



INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(51) International Patent Classification ⁷ : G06F 17/60	A2	(11) International Publication Number: WO 00/52618
		(43) International Publication Date: 8 September 2000 (08.09.00)

(21) International Application Number: PCT/US00/05080

(22) International Filing Date: 29 February 2000 (29.02.00)

(30) Priority Data:
09/260,079 2 March 1999 (02.03.99) US

(71) Applicant: AURIGIN SYSTEMS, INC. [US/US]; 10710 North Tantau Avenue, Cupertino, CA 95014-0717 (US).

(72) Inventors: RIVETTE, Kevin, G.; 2165 Waverley Street, Palo Alto, CA 94303 (US). RAPPAPORT, Irving, S.; 1500 Edgewood Drive, Palo Alto, CA 94303 (US). HOHMANN, Luke; 306 Windmill Park Lane, Mountain View, CA 94043 (US). PUGLIA, David; 17429 East Vineland Avenue, Los Gatos, CA 95030 (US). DEWOLFE, Andrew, S.; 242 Acalanes Drive #11, Sunnyvale, CA 94086 (US). GORETSKY, David; 272 Waverly Street, Sunnyvale, CA 94086 (US). JACKSON, Adam; 1063 Morse Avenue #7-107, Sunnyvale, CA 94089 (US). KUROWSKI, Scott; 1038 Corvette Drive, San Jose, CA 95129 (US). PARK, Brian; 2636 Ponce Avenue, Belmont, CA 94002 (US). RABB, Charles, Jr.; 730 East Evelyn #638, Sunnyvale, CA 94086 (US). ROSENQUIST, Brent; 1668 Kennard Way, Sunnyvale, CA 94087 (US). SCHNITZ, Matthew; 2558 Mardell Way, Mountain View, CA 94043 (US). SMITH,

David, W.; 3 Morning Sun Court, Mountain View, CA 94043 (US). PARADAN, Thierry; 1058 Paintbrush Drive, Sunnyvale, CA 94086 (US). BASHSHUR, Noura; 306 Windmill Park Lane, Mountain View, CA 94043 (US).

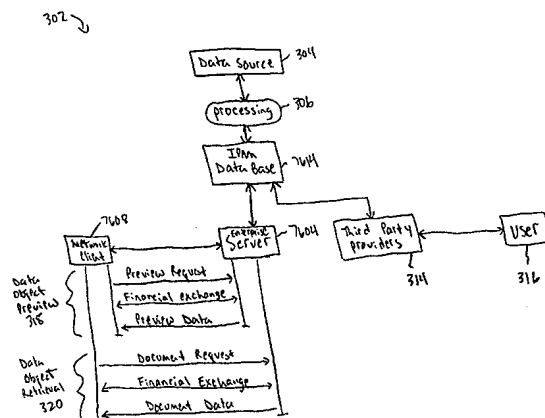
(74) Agents: LEE, Michael, Q. et al.; Sterne, Kessler, Goldstein & Fox P.L.L.C., Suite 600, 1100 New York Avenue, N.W., Washington, DC 20005-3934 (US).

(81) Designated States: AE, AL, AM, AT, AU, AZ, BA, BB, BG, BR, BY, CA, CH, CN, CR, CU, CZ, DE, DK, DM, EE, ES, FI, GB, GD, GE, GH, GM, HR, HU, ID, IL, IN, IS, JP, KE, KG, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, MA, MD, MG, MK, MN, MW, MX, NO, NZ, PL, PT, RO, RU, SD, SE, SG, SI, SK, SL, TJ, TM, TR, TT, TZ, UA, UG, UZ, VN, YU, ZA, ZW, ARIPO patent (GH, GM, KE, LS, MW, SD, SL, SZ, TZ, UG, ZW), Eurasian patent (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European patent (AT, BE, CH, CY, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE), OAPI patent (BF, BJ, CF, CG, CI, CM, GA, GN, GW, ML, MR, NE, SN, TD, TG).

Published

Without international search report and to be republished upon receipt of that report.

(54) Title: INTELLECTUAL PROPERTY ASSET MANAGER (IPAM) FOR CONTEXT PROCESSING OF DATA OBJECTS



(57) Abstract

Context data processing is described herein. One or more contexts are selected. Each context includes one or more attributes, and a plurality of data objects that satisfy the attributes. A list of data objects contained in the selected contexts is displayed. At least some of the data objects in the selected contexts are processed. Such processing may involve generating claim trees, citation trees, and data object families, which may be displayed using hyperbolic trees. In an embodiment, the contexts are groups. In other embodiment, the contexts are each associated with a data object type. In this latter embodiment, the contexts include data objects of their respective data object types.

FOR THE PURPOSES OF INFORMATION ONLY

Codes used to identify States party to the PCT on the front pages of pamphlets publishing international applications under the PCT.

AL	Albania	ES	Spain	LS	Lesotho	SI	Slovenia
AM	Armenia	FI	Finland	LT	Lithuania	SK	Slovakia
AT	Austria	FR	France	LU	Luxembourg	SN	Senegal
AU	Australia	GA	Gabon	LV	Latvia	SZ	Swaziland
AZ	Azerbaijan	GB	United Kingdom	MC	Monaco	TD	Chad
BA	Bosnia and Herzegovina	GE	Georgia	MD	Republic of Moldova	TG	Togo
BB	Barbados	GH	Ghana	MG	Madagascar	TJ	Tajikistan
BE	Belgium	GN	Guinea	MK	The former Yugoslav Republic of Macedonia	TM	Turkmenistan
BF	Burkina Faso	GR	Greece			TR	Turkey
BG	Bulgaria	HU	Hungary	ML	Mali	TT	Trinidad and Tobago
BJ	Benin	IE	Ireland	MN	Mongolia	UA	Ukraine
BR	Brazil	IL	Israel	MR	Mauritania	UG	Uganda
BY	Belarus	IS	Iceland	MW	Malawi	US	United States of America
CA	Canada	IT	Italy	MX	Mexico	UZ	Uzbekistan
CF	Central African Republic	JP	Japan	NE	Niger	VN	Viet Nam
CG	Congo	KE	Kenya	NL	Netherlands	YU	Yugoslavia
CH	Switzerland	KG	Kyrgyzstan	NO	Norway	ZW	Zimbabwe
CI	Côte d'Ivoire	KP	Democratic People's Republic of Korea	NZ	New Zealand		
CM	Cameroon			PL	Poland		
CN	China	KR	Republic of Korea	PT	Portugal		
CU	Cuba	KZ	Kazakstan	RO	Romania		
CZ	Czech Republic	LC	Saint Lucia	RU	Russian Federation		
DE	Germany	LI	Liechtenstein	SD	Sudan		
DK	Denmark	LK	Sri Lanka	SE	Sweden		
EE	Estonia	LR	Liberia	SG	Singapore		

Intellectual Property Asset Manager (IPAM) for Context Processing of Data Objects

Background of the Invention

Field of the Invention

5 The present application is a continuation-in-part of pending application "System, Method, and Computer Program Product for Managing and Analyzing Intellectual Property (IP) Related Transactions," Ser. No. 09/138,368, filed August 21, 1998, which is a continuation-in-part of pending application "Using Hyperbolic Trees to Visualize Data Generated By Patent-Centric and Group-Oriented Data Processing," Ser. No. 07/921,369, filed August 29, 1997, which is a continuation-in-part of allowed application "System, Method, and Computer Program Product for Patent-Centric and Group-Oriented Data Processing," Ser. No. 07/867,392, filed June 2, 1997, all of which are incorporated herein by reference in their entireties.

10 The present invention is generally related to data processing, and more particularly related to, *inter alia*, processing data objects in a variety of contexts.

Related Art

15 Patents are becoming more and more important to a business's success, especially in today's global economy. Patents can be viewed as a new type of currency in this global economy because they grant the holder with a right to exclude others from making, using, or selling the patented technology. In some industries, product turnover is fairly rapid. However, core technology, product features, and markets change at a much slower rate. Accordingly, even in fast-moving industries, patents which cover core technology are very valuable at protecting a company's research and development investment for an extended period of time.

Patents are also valuable as revenue generators. In 1993, for example, the revenue generated from patents by U.S. companies was over \$60 billion. Fred Warshofsky, *The Patent Wars*, John Wiley & Sons, Inc., New York, 1994. These patent revenue dollars are rising each year.

5 Patents are further valuable because they collectively represent a vast technological database. According to Larry Kahaner's book, *Competitive Intelligence*, Simon & Schuster, 1996, "More than 75 percent of the information contained in U.S. patents is never released anywhere else."

10 More and more corporations are recognizing the value of patents. The number of patents applied for and issued to U.S. companies is increasing every year, especially in fast moving industries such as computer software, telecommunications, and biotechnology. Many international companies have also recognized the value of patents. In fact, foreign companies regularly rank among the leaders in issued U.S. patents.

15 Yet, for all the heightened awareness being paid to patents in some quarters, patents remain one of the most underutilized assets in a company's portfolio. This is due, at least in significant part, to the fact that patent analysis, whether for purposes of licensing, infringement, enforcement, freedom to operate, technical research, product development, etc., is a very difficult, tedious, time
20 consuming, and expensive task, particularly when performed with paper copies of patents. Often times, it is a difficult or even impossible task to simply identify relevant patents. Accordingly, detailed patent related analysis is usually not done, or it is done in an ad hoc, unorganized, incomplete, inefficient, and/or ineffective manner.

25 It would be very beneficial to have automated tools that aid users in processing and analyzing patent-related information and non-patent related information for making corporate business decisions.

Summary of the Invention

Briefly stated, an embodiment of the invention is directed to systems, methods, computer program products, and combinations thereof, of context data processing. The embodiment operates by selecting one or more contexts. Each context includes one or more attributes, and a plurality of data objects that satisfy the attributes. A list of data objects contained in the selected contexts is displayed. At least some of the data objects in the selected contexts are processed. Such processing may involve generating hierarchical and or directed acyclic graph data structures to represent relationships among the data objects. These data structures can then be displayed in a variety of well-known techniques including but not limited to hyperbolic trees. Examples of such hierarchical or directed acyclic graph structures include claim trees, citation trees, and data object families, which may be displayed using hyperbolic trees.

In an embodiment, the contexts are groups. In other embodiment, the contexts are each associated with a data object type. In this latter embodiment, the contexts include data objects of their respective data object types. The invention is not limited to these embodiments.

The invention supports the generation of annotations. The invention supports a plurality of annotation types, including document annotations, group annotations, data object type annotations, case annotations, and enterprise annotations. The invention also supports form-based annotations.

In an embodiment, the invention is directed to a system having an intellectual property asset manager (IPAM) and a plug-in manager coupled to the IPAM. The system also includes at least one plug-in coupled to the plug-in manager, and at least one external data processing component coupled to the plug-in. In an embodiment, the external data processing component displays data using at least graphs. In another embodiment, the external data processing

component displays data using at least maps. The plug-in manager has a first application programming interface (API), and the external data processing component each has a second API. The plug-in translates messages from the plug-in manager to the external data processing component to a format conforming to the second API, and translates messages from the external data processing component to the plug-in manager to a format conforming to the first API.

The invention is also directed to assignee name processing. Such processing operates by selecting a normalized assignee name for an entity, identifying name representations of the entity in a data set, and linking the name representations to the normalized assignee name.

Further features and advantages of the invention, as well as various embodiments of the invention, are described in detail below with reference to the accompanying drawings.

Brief Description of the Figures

The present invention will be described with reference to the accompanying drawings, wherein:

FIG. 1 illustrates an example group browser;

FIG. 2 illustrates a flowchart of image skimming according to an embodiment of the invention;

FIG. 3 depicts an event trace corresponding to a preview data object function according to an embodiment of the invention;

FIG. 4 illustrates a patent being a member of a plurality of groups;

FIG. 5 illustrates a data object, such as a document, being a member of a plurality of contexts;

FIG. 6 illustrates a corporate document window according to an embodiment of the invention;

FIG. 7 illustrates filtering in a corporate document window according to document kind;

FIG. 8 illustrates filtering in a corporate document window according to security class;

5 FIG. 9 illustrates a new corporate document kind dialog according to an embodiment of the invention;

FIGS. 10 and 11 illustrate a new security class dialog according to an embodiment of the invention;

10 FIG. 12 illustrates a new corporate document dialog according to an embodiment of the invention;

FIGS. 13-15 illustrate the addition of a corporate document to the IPAM system according to an embodiment of the invention;

FIG. 16 illustrates a context browser according to an embodiment of the invention;

15 FIG. 17 illustrates a type table according to an embodiment of the invention;

FIG. 18 illustrates a type document xref table according to an embodiment of the invention;

FIG. 19 illustrates an example type table;

20 FIG. 20 illustrates an example type document xref table;

FIGS. 21 and 22 are used to describe a data object preview function according to an embodiment of the invention;

FIG. 23 is a table illustrating the differences between different annotation types;

25 FIGS. 24-30 illustrate various types of document annotations according to embodiments of the invention;

FIG. 31 illustrates an annotation pane according to an embodiment of the invention;

FIG. 32 illustrates a flowchart representing the operation of the invention when creating an annotation that is linked to a data object;

FIG. 33 illustrates a flowchart representing the operation of the invention when creating a group or type annotation;

5 FIGS. 34-36 illustrate search GUIs according to embodiments of the invention;

FIG. 37 illustrates an example claim tree;

FIG. 38 illustrates a flowchart reflecting the operation of the invention when generating and utilizing a claim tree;

10 FIGS. 39-41 are used to describe a display claim branch function of the present invention;

FIG. 42A depicts a menu applicable to claim trees;

FIG. 42B illustrates node state information that is stored in nodes of hyperbolic trees according to an embodiment of the invention;

15 FIG. 43 illustrates an example patent citation tree;

FIG. 44 depicts a patent ref table according to an embodiment of the invention;

FIG. 45 illustrates a non-patent ref table according to an embodiment of the invention;

20 FIG. 46 illustrates an example patent ref table;

FIG. 47 illustrates an example non-patent ref table;

FIG. 48 illustrates a data object family table according to an embodiment of the invention;

FIG. 49 illustrates an example data object family table;

25 FIGS. 50A and 50B depict a table related to relationship types supported by embodiments of the present invention;

FIG. 51 illustrates an example patent family chronology;

FIG. 52 illustrates an example data object family table corresponding to the patent family chronology of FIG. 51;

FIG. 53 illustrates an example assignee technology patent family;

FIG. 54 illustrates an example data object family table corresponding to the assignee technology patent family of FIG. 53;

FIG. 55 illustrates a flowchart representing the operation of embodiments of the invention when generating relationship tables;

FIG. 56 is used to describe example sources of relationship data utilized by embodiments of the invention;

FIG. 57A is a flowchart representing the operation of the invention when analyzing relationship information;

FIG. 57B is a flowchart representing the operation of the invention when computing closure of a data object family;

FIG. 58 is a diagram illustrating the operation of the enterprise server when interacting with third party components and tools;

FIG. 59 is a flowchart illustrating the operation of the enterprise server when interacting with third party components/tools;

FIG. 60 is a more detailed block diagram of the enterprise server interacting with third party components/tools;

FIG. 61 is an event trace representing the operation of the enterprise server when interacting with third party components;

FIGS. 62A and 62B are a flowchart corresponding to the event trace diagram of FIG. 61;

FIG. 63 is an example data flow diagram relating to the operation of the enterprise server when interacting with a third party component that represents data using landscapes or maps;

FIG. 64 is a flowchart representing an example operation of a third party component interacting with the enterprise server;

FIGS. 65 and 66 represent example displays of data represented as maps or landscapes;

FIG. 67 is an example display of data represented using a graph with dynamically adjustable axis;

FIG. 68 is an event trace of the enterprise server operating with a specific third party component/tool;

5 FIGS. 69-73 depict various configurations and distributions of components and functionality supported by the present invention;

FIG. 74 is a flowchart representing the operation of embodiments of the invention when performing assignee name processing;

10 FIG. 75 illustrates another context browser according to embodiments of the invention;

FIG. 76 is a block diagram of the enterprise server according to embodiments of the invention;

FIG. 77 is a flowchart representing the operation of the invention when performing context processing according to embodiments of the invention.

15 FIGS. 78A and 78B illustrate free text annotations according to an embodiment of the invention;

FIG. 79 illustrates an example form-based annotation;

FIG. 80 illustrates another example formed based annotation;

20 FIG. 81 is a data flow diagram used to describe form-based annotations according to embodiments of the invention;

FIGS. 82, 83A, and 83B are flow charts representing the operation of form-based annotations according to embodiments of the invention;

FIG. 84 is a flow chart representing the operation of the invention when inputting an arbitrary data object;

25 FIG. 85 is a data flow diagram supporting the flow chart of FIG. 84;

FIG. 86 is a block diagram of an example computer system useful for implementing modules of the invention;

FIG. 87 illustrates an example patent claim tree module;

FIG. 88 illustrates an example landscape or map tool operable with embodiments of the invention;

FIG. 89 illustrates an example search tool operable with embodiments of the invention;

5 FIG. 90 illustrates an example export operation; and

FIGS. 91-98 are used to illustrate an example plug-in operation of the invention with a third party tool.

10 In the following text, reference is sometimes made to existing patents. Also, some of the figures reference or illustrate existing patents. For illustrative purposes, information from and/or about these patents has sometimes been modified or created in order to support the particular examples being discussed. Accordingly, the information provided herein about these existing patents should be considered to be fictional unless verified through comparison with copies of the actual patents that are available from national patent offices.

Detailed Description of the Preferred Embodiments

Overview of the Invention

The present invention is directed to systems, methods, computer program products, and components and combinations thereof, for processing data objects. Such processing includes, but is not limited to, acquiring data objects, organizing data objects, storing data objects, visualizing data objects, displaying data objects, manipulating data objects, annotating data objects, processing data contained in data objects, etc. The invention is also directed to combinations of the modules and functions described herein, and also described in U.S. patent applications "System, Method, and Computer Program Product for Patent-Centric and Group-Oriented Data Processing," Ser. No. 08/867,392, "Using Hyperbolic Trees to Visualize Data Generated by Patent-Centric and Group-Oriented Data Processing," Ser. No. 08/921,369, "System, Method and Computer Program Product for Managing and Analyzing Intellectual Property (IP) Related Transactions," Ser. No. 09/138,368, herein incorporated by reference in their entireties.

In embodiments of the invention, the data objects include, but are not limited to, patents and patent-related data objects. More generally, the data objects include, but are not limited to, intellectual property related data objects. The data objects also include, but are not limited to, data objects of past, present, or potentially future interest to corporate entities. For illustrative purposes, the invention is sometimes described in terms of these embodiments. However, the invention is not limited to these embodiments. The invention is applicable to data objects of any type and subject matter. Accordingly, description of the invention in terms of these embodiments is provided for illustrative purposes only, and is not limiting.

5 The invention may be used with immutable data objects (i.e., data objects that cannot be modified), such as but not limited to data objects produced by government agencies and similar institutions. For example, an issued patent is generally considered an immutable data object since it is preferred that it not be modified. An executed contract is another example of a document that can be immutable. However, the invention is not limited to these embodiments. The invention can operate with mutable data objects, and/or data objects produced by private entities.

10 In embodiments of the invention, the data objects include documents, such as textual documents, image documents, and combinations thereof. For illustrative purposes, the invention is sometimes described in terms of these embodiments. However, the invention is not limited to these embodiments. The invention is applicable to data objects of any type and form, including but not limited to video, audio, sensory, multimedia, tactile, computer programs, links, etc., and combinations thereof. Accordingly, description of the invention in terms of these embodiments is provided for illustrative purposes only, and is not limiting.

15 In some embodiments, the present invention is intended to aid a corporate entity with planning over its entire enterprise. Accordingly, the present invention is sometimes referred to as an enterprise system and method (or an enterprise server). In some embodiments, the present invention is intended to aid a corporate entity in developing business-related strategies, plans, and actions. Accordingly, the present invention is sometimes referred to as a business decision system (BDS). In some embodiments, the present invention is intended to aid a corporate entity with managing intellectual property assets. Accordingly, the present invention is sometimes referred to as an intellectual property asset manager (IPAM). In any of these embodiments, the invention can run in a single user or a multi-user mode of operation.

Image Skimming

The invention supports an image skimming function. In accordance with the image skimming function, preferably, the first image page of each document in a collection of documents is displayed in succession pursuant to appropriate user command.

5 The invention is not limited to this embodiment. In particular, the skimming feature of the invention is not limited to document images. The skimming feature enables one to view or otherwise obtain information about a data object in a form that is consistent with that data object. For example, and without limitation, if a group contained a set of video clips, the skimming pane
10 might allow for a video clip to be displayed/played directly from that window.

FIG. 1 illustrates an example user interface 102 according to embodiments of the invention. The user interface 102 is called a console or a browser.

A group hierarchy is shown in a group pane 104 of the browser 102. A
15 group is a data structure that includes a collection of data objects, such as documents. The data objects in a group typically follow a common theme or characteristic. For example, a first group may include patents and other documents that map to a product being manufactured and sold by a company. A second group may include patents and other documents that map to a product
20 being considered for future manufacture and sale by a company. A third group may include patents and other documents related to a research product. A group may contain other groups, or may be included in one or more groups. Accordingly, in embodiments of the invention, groups are hierarchically organized. Groups are further described in U.S. Patent Application "System,
25 Method, and Computer Program Product for Patent-Centric and Group-Oriented Data Processing," Ser. No. 08/867,392, herein incorporated by reference in its entirety.

As noted above, a group hierarchy is shown in a group pane 104 of the browser 102. In the example of FIG. 1, a group "771 docs" is selected, as

indicated as 105. A list of data objects contained in the selected group 105 is displayed in a contents pane 106. In the example of FIG. 1, a document "EP 0815816A1" is selected, as indicated as 107. Information pertaining to the selected data object 107 is displayed in a data object pane 110. Such information, for example, can include a summary and/or bibliographic information of the selected data object 107 (see the summary tab 114 in the data object pane 110), the text of the selected data object 107, and/or image data relating to the selected data object 107 (see the image tab 112 in the data object pane 110). Other information pertaining to the selected data object 107 is also accessible via the data object pane 110, as further described below.

The contents pane 106 includes navigational buttons 116, 118 which preferably operate relative to the selected data object 107 in the contents pane 106. When selected, button 116 (represented as a down arrow) preferably causes information corresponding to at least the first image page of the next data object in the list of data objects in the contents pane 106 to be retrieved and displayed in the image tab 112 of the data object pane 110. The button 118 (preferably represented as an up arrow), when selected, preferably causes data pertaining to at least the first image page of the prior data object in the list of data objects in the contents pane 106 to be retrieved and displayed in the image tab 112 of the data object pane 110.

According to the image skimming function of the present invention, the first image page of successive documents listed in the contents pane 106 is displayed by repeatedly clicking the navigation buttons 116 and 118. Specifically, to move down the list of data objects displayed in the contents pane 106, the down arrow button 116 is pressed by the user. To move up the list of data objects listed in the contents pane 106, the up arrow 118 is repeatedly pressed by the user.

In an embodiment of the invention, the data objects listed in the contents pane 106 represent patents or documents related to patents (including, but not

limited to, post-issuance documents). In this embodiment, the image skimming feature of the present invention is analogous to a manual search of patents in the shoes of the patent search room, and the collection of documents being searched is analogous to one or more electronic patent shoes. The collection may be one or more groups, for example. During such manual searches, sometimes called "patent shoe searching," practitioners often quickly thumb through the patents in a patent shoe by looking at the first pages of the patents. The image skimming feature of the present invention enables an operator to quickly electronically scan over the first image pages of the patents listed in the contents pane 106, thereby emulating a manual search through the shoes in the patent office search room.

The present invention, however, represents a significant improvement over a manual patent shoe search. These advantages generally pertain to perspective and scope. Specifically, the present invention provides both a global view of the subject matter being searched, and a detailed view of the subject matter being searched. The global view is represented by the list of patents and related patent data objects listed in the contents pane 106. In particular, the contents pane 106 provides the user with an overview of the data objects that is the subject of the analysis.

The detailed view is represented by the data object pane 110, which provides the user with detailed information regarding the currently selected patent from the contents pane 106.

The global view (i.e., the contents pane 106) and the detailed view (i.e., the data object pane 110) are simultaneously displayed for review by the user. Thus, the user can simultaneously electronically view the list of the patents being searched, and the first page of the currently selected patent. Also, the user can immediately modify the focus of the search by selecting any patent listed in the contents pane 106. Such "random access" is not possible with manual patent shoe searching.

Also, the group pane 104 provides an even higher level view of the subject matter being searched.

Further, while conducting the search, the user can easily and effectively create electronic annotations to capture his or her analysis. The annotation features of the present invention are further described below.

FIG. 2 illustrates a flowchart 202 representing the operation of the present invention when performing the image skimming function.

In step 206, the user selects a collection of data objects. The collection of data objects can be selected using any of the features and functions of the present invention described herein. For example, the user can conduct a search wherein the data objects identified by the search represents the collection of step 206. Alternatively, the user can select one or more groups, wherein the data objects contained in the selected groups correspond to the collection of step 206. Still alternatively, the user can select one or more document types, wherein all data objects of the selected document types correspond to the collection of data objects of step 206. A document collection/group can be selected through any combination of searches, attributes, document types, etc., all working together (e.g., "Create a group that consists only of license agreements (document type) containing the text "digital video" (keyword search) whose expiration date is between 1/1/1999 and 12/31/1999 (document attribute)).

In step 208, the data objects in the selected collection are displayed in the contents pane 106.

In step 210, one of the data objects listed in the contents pane 106 is selected. Such selection may be manually performed by the user. Alternatively, by default, one of the documents may be preselected, such as the first document in the list, or the last document in the list. This default operation may be user controllable/adjustable.

In step 212, information pertaining to the selected data object is retrieved. Referring to FIG. 76, an attempt to retrieve such information is first made to

local databases 7612. If image information of the selected document is not contained in the local databases 7612, then the network client 7608 interacts with the enterprise server 7604 to retrieve the image data from the remote databases 7614. Additional details pertaining to the enterprise server 7604 and the network client 7608, as well as other component shown in FIG. 76, is contained in U.S. Patent Application "System, Method, and Computer Program Product for Patent-centric and Group-oriented Data Processing," Ser. No. 08/867,392, incorporated herein by reference in its entirety.

In step 214, information pertaining to the selected document is displayed in the data object pane 110. Preferably, information corresponding to at least the first image page of the selected document is displayed in the image tab 112 of the data object pane 110.

In step 216, the user enters a navigational command. Specifically, the user either presses the down arrow button 116 or the up arrow button 118.

If the user pressed the up arrow button 118, then information pertaining to at least the first image page of the prior document listed in the contents pane 106 is retrieved and displayed, as indicated by steps 218, 220, and 222. If, instead, the user selected the down arrow button 116, then information pertaining to at least the first image page of the next data object listed in the contents pane 106 is retrieved and displayed. This is represented by steps 224, 226, and 228. If neither the up arrow button 118 nor the down arrow button 116 was pressed by the user, then the system processes the command that was issued as appropriate, as represented by step 230.

Image skimming is further described in U.S. Patent Application "System, Method, and Computer Program Product for Patent-centric and Group-oriented Data Processing," Ser. No. 08/867,392, incorporated herein by reference in its entirety.

Context Processing

The present invention supports context processing. A context is a data structure comprising one or more attributes. The context also comprises data objects that satisfy, follow, relate to, and/or have the context's attributes. Contexts are further described below.

5 ***Group-Centric Processing***

The invention supports a variety of contexts. For example, a group is a context.

10 As noted above, a group is a data structure that includes a collection of data objects, such as documents. The documents in a group typically (but not exclusively) follow a common theme or characteristic. Groups are typically (but not exclusively) based on subject matter. The data objects in a group relate to (or map to) the subject matter associated with the group. For example, a first group may include patents and other documents that map to a product being manufactured and sold by a company. A second group may include patents and
15 other documents that map to a product being considered for future manufacture and sale by a company. A third group may include patents and other documents related to a research product.

 Accordingly, groups are a context for organization, visualization, analysis, and other processing.

20 Groups are further described in U.S. Patent Application "System, Method, and Computer Program Product for Patent-centric and Group-oriented Data Processing," Ser. No. 08/867,392, incorporated herein by reference in its entirety.

Non-Group-Centric Processing

The invention supports contexts other than groups. Context processing performed by the present invention that does not involve groups is herein referred to as non-group-centric processing.

5 The invention supports a variety of non-group-centric contexts. For example, a context can be created as a result of processing performed by the system. A context can be created from documents identified by a forward citation tree operation. Similarly, a context can be created from a backward citation tree operation. Another context can be created by searching all groups and/or data object types in the system containing a given document, or having
10 some attribute. In that instance, the context would include all hits from that search.

A particular context supported by the invention is a data object type context. This context is described below.

Data Object Type Processing

15 According to the invention, each data object has one or more data object kinds. A data object kind represents a classification of a data object. For example, data object kinds include license agreements, non-disclosure agreements, patents, tax forms, personnel evaluations, software programs, user
20 manuals, technical manuals, copyright registration forms, audio clips, movies, multimedia clips, data object links, annotations, and any other data object or document classification of interest to the user.

Data object kinds represent contexts for organization, visualization, analysis, and other processing. For example, it may be useful to analyze all
25 license agreements in which a corporate entity is a party. In this example, the user selects the data object kind "license agreements." All license agreements associated with the corporate entity are selected and may be analyzed by the user. All other information perhaps pertaining to the corporate entity, but not

representing license agreements, are excluded from analysis. Thus, the present invention facilitates the organization, visualization, analysis, and other processing of information of interest to the user.

It is illustrative to note the differences between a group and a data object kind. A group is defined by its attributes, and the documents and other data objects in the group share or follow such attributes. In contrast, a data object kind represents a classification of a data object. The data objects associated with the data object kind may have nothing in common with each other except that they are the same kind of data object.

A given data object, such as a patent, may exist in a plurality of groups. This is illustrated in FIG. 4, where a given patent is contained in a bill of materials group, a user find group, an inventor group, a corporate entity group, etc.. Similarly, a data object may exist in a plurality of contexts. This is illustrated in FIG. 5, for example, where a given data object is included in a group context, an author context, a publisher context, and a topic context.

Corporate Document Browser

FIG. 6 illustrates an example corporate document window or browser 602. Corporate documents are listed in the corporate document window 602. Preferably, corporate documents are defined as non-patent documents. In alternative embodiments, however, corporate documents may include both patents and non-patent documents.

Each corporate document includes a document ID, a title, a data object kind, and a security class. This information is listed in columns 604, 606, 608, and 610, respectively, of the corporate document window 602.

The invention preferably includes various filters to enable users to limit the documents that are listed in the corporate document window 602. The example corporate document window 602 shown in FIG. 6 includes a security

class filter 614 and a data object kinds filter 616. As shown in FIG. 7, the user can select one or more data object kinds for viewing. If, for example, the user selects "contract" in the data object kind filter 616, then only documents having the data object kind of "contract" are listed in the corporate document window 602.

The security class filter 614 allows the user to select one or more security classes. Only documents of the selected security classes are listed in the corporate document window 602. As shown in FIG. 8, for example, if the user selected in the security class filter 614 the class "class 5," then only documents of security class "class 5" are listed in the corporate document window 602.

Preferably, access to documents are restricted/managed/maintained by security information. A user can only access/browse those documents that are consistent with his security information.

The browsers of the invention support inverse functions. An inverse function is one that performs an operation that is the inverse of another operation. For example, browsing a group displays all of the documents in the group. The inverse is browsing all of the groups that contain the document. Many of the browser functions described herein include inverse functions. The inverse functions available for any given browser varies and is implementation dependent.

Data Object Kinds

As noted above, a data object kind is a classification of a document. For example, all license agreements preferably have a data object kind of "license agreement." All non-disclosure agreements preferably have a data object kind of "non-disclosure agreement." All patents preferably have a data object kind of "patent." The level of granularity of the data object kinds is implementation and

application dependent. For example, there may be data object kinds for "U.S. patent," "Japanese patent," "Canadian patents," etc.

A data object may have more than one data object kind. For example, a license agreement may have the data object kind of "license agreement" and "contract."

Typically, the data object kind supported by a system varies over time. Also, typically, different systems will have different data object kinds. Generally, the data object kinds defined and supported by a system correspond to the needs of the users of the system. For example, the data object kinds associated with a patent department may include patent, patent application, license agreement, non-disclosure agreement, and patentability opinion. The data object kinds associated with an accounting department may include payroll statement, invoice, account receivable, license agreement, and budget.

Generally, data object kinds can be created for any criteria or attribute of past, present, or future interest. Data object kinds can be arbitrarily created.

FIG. 9 illustrates a new corporate data object kind window 902, which is displayed when the user elects to create a new data object kind. Preferably, the user enters a name of the data object kind, a description of the data object kind, and an owner of the data object kind. A new data object kind having such name, description, and owner information is created upon pressing the OK button 904.

Security Classes

Preferably, each data object has a security class. Many data objects can have the same security class.

A security class specifies the rights that users have. In other words, a security class specifies user access rights. When a security class is assigned to a document, then access to that document is controlled by the user access rights specified in the security class. Access to the document can be modified by either

changing the access rights associated with the security class, or assigning a new security class to the document. For example, suppose that a security class A gives read and write privileges to a user 1. If document X is assigned security class A, then user 1 has read and write privileges to document X.

5 FIG. 10 illustrates a new security class window 1002, which is displayed when the user wishes to create a new security class. Preferably, as indicated by the security class tab 1004 in the new security class window 1002, a security class has a name, description, and owner. Referring to FIG. 11, as indicated by the permissions tab 1006, a security class specifies the access privileges of users.
10 A new security class is created when the name, description, owner, and user access information is entered and the OK button 1008 is pressed.

Corporate Documents

 In an embodiment of the invention, a corporate document is a data object in the system. It is noted that the following description is applicable to any type
15 of data object (document or otherwise).

 FIG. 12 illustrates a new corporate document window 1202 which is used to create a new corporate document. The new corporate document window 1202 includes a file path field. The path to electronic information representative of the document or other data object that is being added to the system as a corporate
20 document is specified in the file path field. This path can be specified via well known browsing techniques, which is initiated by pressing the browse button 1206. This is just one mechanism that serves to identify the location of a document. The invention is not limited to this embodiment. Other mechanisms could be used, such as but not limited to the use of a URL (Universal Resource
25 Locator).

 The new corporate document window 1202 also includes a document ID field in which the document ID of the new corporate document is displayed.

Preferably, the document ID is unique among all corporate documents and data objects, and is manually inputted. Alternatively, it is automatically generated by the system.

5 The new corporate document window 1202 also includes a title field, in which the user can enter a title of the new corporate document. A data object kind and a security class for the new corporate document can be entered in fields 1212 and 1214. If the user wishes to create a new data object kind, the user can press the new button 1216, in which case the new corporate data object kind window 902 (FIG. 9) is displayed. If the user wishes to create a new security class, the user presses the new button 1218, in which case the new security class window 1002 (FIGS. 10 and 11) is displayed. The user can also enter additional bibliographic information pertaining to the corporate document, such as author, source, disclosure date, and publication date. These fields are available in the additional bibliographic fields tab 1222 (See FIG. 14).

10
15 FIGS. 13 and 14 illustrate an example where a new corporate document titled "License Agreement Between ABC and EFG" is being created. FIG. 15 illustrates the corporate document window 1302, which displays corporate documents in the system. Note that the new corporate document titled "License Agreement Between ABC and EFG" is listed at 1502.

20 *Data Object Type Databases*

Example database tables useful for implementing data object kinds as described above are considered in this section.

FIG. 17 illustrates a type table 1702. The type table 1702 stores information for data object types supported by the system. The type table 1702 includes a record for each data object type defined and supported in the system. Each record of the type table 1702 includes a type_ID attribute corresponding to the unique identifier of the data object type, a name attribute containing the name

of the data object type, a description attribute containing a description of the data object type, and an owner attribute specifying an owner of the data object type.

FIG. 18 illustrates a type document xref table 1802. The type document xref table 1802 stores information that identifies the data objects in each data object type supported by the system. The type document xref table 1802 includes a record for each data object in each data object type supported by the system. Each record of the type document xref table 1802 includes a type_ID attribute that identifies a data object type, and a document_ID attribute that identifies a data object that is in the data object type.

An example of the type table 1702 and the type document xref table 1802 is illustrated in FIGS. 19 and 20. FIG. 19 illustrates an example type table 1902. The type table 1902 includes a data object type called "automation" that has a type_ID of T1. The type table 1902 also includes information pertaining to a data object type called "license agreement" that has a type_ID of T2. FIG. 20 illustrates an example type document xref table 2002. This example type document cross ref table 2002 indicates that contract 1 and contract 3 are of data object type "automation," and contract 1 and contract 2 are of data object type "license agreement."

Context Browser

FIG. 16 illustrates an example context browser 1602, which is similar to the group browser 102 shown in FIG. 1. The context browser 1602 includes a kind pane 1604 in which the data object kinds supported by the system are listed. In some embodiments, the data object kinds are hierarchically organized. For example, a license agreement kind and a non-disclosure agreement kind may be children of a contracts kind. A U.S. patents kind and an Australian patents kind may be children of a patents kind. The hierarchical organization of data object kinds is indicated in the kind pane 1604.

A content pane 1606 list the data objects contained in a data object kind selected in the kind pane 1604. Annotations are listed in an annotations pane 1608 (annotations are further described below). Information pertaining to a data object selected in the contents pane 1606 is displayed in a data object pane 1610. Such information includes any information pertaining to the selected data object, such as summary or bibliographic information in the summary tab 1612, text information pertaining to the selected document in the text tab 1614, image information pertaining to the selected data object in the image tab 1616, audio information associated with the selected document in the audio tab 1618, and video information associated with the selected document in the video tab 1620. Other information pertaining to the selected data object, if any, is also accessible via the data object pane 1610 via additional tabs (not shown in FIG. 16).

FIG. 75 illustrates another example context browser 7502. The context browser 7502 includes a kind pane 7504, a contents pane 7506, and a data object pane 7508. These panes are similar to those shown in FIG. 16.

Operation of Context Processing

Context processing according to embodiments of the invention is illustrated in a flowchart 7702 in FIG. 77. In step 7704, one or more contexts are selected. For example, one or more groups and/or one or more data object types are selected. Each of the selected contexts include any number of data objects.

In step 7706, the data objects in the selected contexts are processed. Such processing includes any combination of the functions described herein, and/or the functions described in U.S. Patent Applications titled "System, Method, and Computer Program Product for Patent-centric and Group-oriented Data Processing," Ser. No. 08/867,392, "Using Hyperbolic Trees to Visualize Data Generated by Patent-centric and Group-oriented Data Processing," Ser. No.

08/921,369, and/or "System, Method, and Computer Program Product for Managing and Analyzing Intellectual Property (IP) Related Transactions," Ser. No. 09/138,368, all of which are herein incorporated by reference in their entireties. Such processing may be automatic, manual, or combinations thereof.

5 ***Data Object Preview***

FIG. 76 illustrates a block diagram of an embodiment of the present invention. An enterprise server 7604, which is also referred to as an intellectual property asset manager (IPAM) and a business decision system (BDS), performs processing on data objects as described herein. Information on such data objects is stored in IPAM databases 7614. Such information includes bibliographic information, textual information, image information, video information, audio information, and other information associated with the data objects.

10 Preferably, users interact with the enterprise server 7604 and the IPAM databases 7614 via network clients 7608 and web clients 7606. Network clients 7608 interact directly with the enterprise server 7604 (although there may be networks between the network clients 7608 and the enterprise server 7604). Web clients 7606 interact with the enterprise server 7604 via web server 7610.

15 Preferably, network clients 7608 include local databases 7612. Web clients 7606 may also include local databases. The local databases 7612 store information on data objects. Preferably, the information contained in the local database 7612 is a subset of the information contained in IPAM databases 7614. For example, the local database 7612 may only include information on a limited number of data objects. Also, for any given data object, the local database 7612 may only store some information pertaining to the data object. For example, the database 7612 may only store bibliographic or summary information pertaining to the data object. The database 7612 may not store more detailed information

20

25

pertaining to the data object, such as textual information, image information, video information, audio information, etc.

Further information pertaining to the enterprise server 7604, the network client 7608, the web client 7606, the web server 7610, and databases 7614 and 7612 are found in U.S. Patent Application "System, Method, and Computer Program Product for Patent-centric and Group-oriented Data Processing," Ser. No. 08/867,392, incorporated herein by reference in its entirety.

FIG. 21 illustrates a browser 2102 (also called a console), which may be a group browser or a context browser. As described above, the contents pane 2108 lists documents that are contained in the group or kind selected in the group/kind pane 2106.

The data object pane 2104 illustrates information pertaining to the document selected in the contents pane 2108. Preferably, the data object pane 2104 displays summary or bibliographic information pertaining to the document selected in the contents pane 2108. This is the case in the example FIG. 21.

The user can elect to display other information pertaining to the selected document by selecting another tab in the data object pane 2104, such as the image tab 2110 shown in FIG. 21. However, such additional information pertaining to the selected document may not be present in the local database 7612. The invention supports a document or data object preview function for addressing such instances.

Specifically, in cases where requested information pertaining to a document or other data object selected in the contents pane 2108 is not present in the local databases 7612, the invention displays a preview button 2106. When the user presses the preview button 2106, the network client 7608 or the web client 7606 interacts with the enterprise server 7604 to retrieve at least a portion of the additional information pertaining to the selected data object. Preferably, the text of the selected document is transferred from the IPAM databases 7614

to the network client 7608 or the web client 7606. This text is then displayed in a preview window 2202 for review by the user (see FIG. 22).

In other embodiments, additional information is transferred from the IPAM databases 7614 to the network client 7608 or the web client 7606. Such information may include image information, audio information, video information, or any other information pertaining to the selected data object that is stored in the IPAM databases 7614. As shown in FIG. 22, such information can be displayed by selecting the appropriate tab in the preview window 2202.

In some embodiments, if the requested information on the selected document is not present in the IPAM databases 7614, the enterprise server 7604 operates to obtain the information from an external source. For example, the enterprise server may interact with a third party information provider, such as Lexis-Nexis, or may interact with some other source, such as a human operator, a company database (such as a personnel database, a finance database, etc.), etc.

The user reviews the information in the preview window 2202. Preferably, this information is available only on a temporary basis, and it is not stored in the local databases 7612. If the user wishes to store the information in the local databases 7612, then the user presses an order button 2206 or issues a similar command. In response to the selection of the order button 2206, the enterprise server 7604 retrieves any additional information pertaining to the selected document from the IPAM databases 7614 or from some other source, and transfers such information to the network client 7608 or the web client 7606. This information, as well as information previously provided, is stored in the local databases 7612.

The processing pertaining to the data object preview function of the present invention is illustrated in FIG. 3. Specifically, FIG. 3 illustrates the interaction between the network client 7608 and the enterprise server 7604 when performing the data object preview function 318. The network client 7608 sends a preview request message to the enterprise server 7604. The preview request

message is sent by the network client 7608 to the enterprise server 7604 when the user pressed the preview button 2106. The preview request message includes information that identifies the data object selected in the contents pane 2108. The preview request message also indicates the type of data that is requested. For example, in some embodiments, only text data pertaining to the selected data object is requested. In other embodiments, other types of data is requested, such as image data, audio data, video data, or any other or all data pertaining to the selected data object. In other embodiments, a given type of data is always requested and provided, such as textual data, in which case a preview request message does not include an indication of the type of data that is requested.

Optionally, the network client 7608 and the enterprise server 7604 perform a financial exchange whereby funds are transferred from the network client 7608 to the enterprise server 7604. The financial exchange is performed in those cases where the preview function is available only as a pay-for-used service. In the financial exchange, the funds are electronically or otherwise transferred from the network client 7608 to the enterprise server 7604. The invention also includes other economic models, such as but not limited to an unlimited service plan, where for a given fee the user can download any number of documents, and a hybrid plan where for a fee the user can download a given number of documents. Once this limit is exceeded, the user pays on a per-document rate.

The enterprise server 7604 retrieves from the IPAM database 7614 the requested data pertaining to the selected data object, and transfers this data to the network client 7608. The IPAM database 7614, in one embodiment, obtains such data from a data source 304. The data is then processed (indicated by oval 306), as necessary, and then stored in the IPAM database 7614. The data source 304, for example, may be a Patent Office, departments of a corporate entity, third party information providers (as discussed above), etc., and/or may result from

data processing performed by automatic means, manual means, or combinations thereof.

Upon receipt of the preview data by the network client 7608, the network client 7608 displays the preview data in the preview window 2202, as described above.

The operations performed by the present invention when the user requests the retrieval of a data object is also represented in FIG. 3 as a data object retrieval function 320. In accordance with such processing, the network client 7608 transfers a document request to the enterprise server 7604. The document request was generated by the network client 7608 when the user pressed the order button 2206 in the preview window 2202 (FIG. 22). The document requests includes information that identifies the selected data object in the contents pane 2108 and also identifies the type of data that is requested (i.e., text, image, audio, video, etc.). In cases where the data object request functionality is offered only as a pay-per-use service, the network client 7608 and the enterprise server 7604 perform a financial exchange operation in which funds are electronically or otherwise transferred from the network client 7608 to the enterprise server 7604. The enterprise server 7604 retrieves the requested information from the IPAM database 7614 or other source, and transfers such information to the network client 7608. The network client 7608 then preferably stores this information in its local databases 7612 (FIG. 76) for later access by the user.

Annotation

The present invention supports a variety of annotation types. An annotation type is defined by two attributes. First, by whether annotations of the annotation type are attached or linked to data objects, or portions of data objects. Second, by the scope of annotations of the annotation type. The scope defines the extent to which the annotations are visible or accessible. FIG. 23 is a table

2302 that summarizes the attributes of the annotation types of the present invention.

Annotations are further described in U.S. Patent Nos. 5,623,681; 5,623,679; and 5,806,079, all of which are herein incorporated by reference in their entireties.

Document Annotations

A document annotation is attached or linked to a selected portion of a data object contained in a group. The scope of a document annotation is the group in which the linked data object is contained, and that was active or selected at the time that the document annotation was created. Document annotations are accessible only when their respective groups are active.

A document annotation may be linked to multiple data objects. Also, a document annotation may be unlinked from any or all linked data objects. When a document annotation is unlinked from all linked data objects, it is called a floating, free, or unlinked annotation (as opposed to a linked annotation). Unlinked annotations maintain their original scope, and operate just like linked annotations, except that they have no links to data objects to follow. Unlinked annotations can be linked to data object portions through appropriate user command. It is noted that this description pertaining to document annotations is also applicable to other linkable annotation types described herein.

For example, assume that a document annotation is linked to a document D1. Document D1 is contained in groups G1 and G2. However, at the time that the document annotation was created, group G1 was active. That is, group G1 was selected in the group pane 104 of the group browser 102 (see FIG. 1) at the time that the annotation was created. In this case, the document annotation is linked to document D1, and has a scope of group G1. The document annotation

is only accessible when group G1 is active. The document annotation is not accessible when group G2 is active, even though document D1 is in G2.

Patent Annotations

5 The invention supports a variety of types of document annotations. For example, the invention supports patent annotations. Consider FIG. 24, which displays a summary of U.S. Patent No. 5,668,742 in a data object pane 2406 of browser 2402. Upon appropriate user command, the text of U.S. Patent No. 5,668,742 is displayed in a text window 2502 shown in FIG. 25. According to the invention, a patent annotation is created by selecting a portion of a patent, 10 such as the selected portion 2504 shown in FIG. 25. Such operation results in creating a subnote 2509 in a note window 2508 that is linked to the selected portion 2504 of U.S. Patent No. 5,668,742.

User comments can be entered into the subnote 2509. In an embodiment, location information 2511 is stored and displayed proximate to the subnote 2509 15 (this is applicable for all annotation types described herein). Such location information 2511 indicates the location of the selected portion 2504 in the data object, *i.e.*, U.S. Patent No. 5,668,742.

Notes are listed in the annotation pane 2510. For example, in the current example, the newly created note is listed in the annotation pane 2510 as entry 20 2512.

Corporate Document Annotations

25 The invention also supports corporate document annotations, in which any type of document can be annotated. For example, in FIG. 28, summary information pertaining to a corporate document selected in a contents pane 2404 is displayed in the data object pane 2406. Image information pertaining to the

selected document is displayed in an image window 2902 upon execution of appropriate user commands, as shown in FIG. 29. As indicated in FIG. 30, a portion 3002 of the image in image window 2902 has been selected. This operation causes a subnote 3006 to be created in a note window 3004. The subnote 3006 is linked to the selected portion 3002. The user can enter information or comments pertaining to the selected portion 3002 in the subnote 3006 (that is, the user can annotate the selected corporate document).

Image Annotations

The invention supports image annotations, wherein images can be annotated. For example, in FIG. 26, image information pertaining to U.S. Patent No. 5,668,742 is displayed in the data object pane 2406 of the browser 2402. Upon appropriate user command, such image information is displayed in larger form in an image window 2702 shown in FIG. 27. As indicated in FIG. 27, the user has selected a portion of the image. This selected portion is indicated as 2704. Upon such selection, a subnote 2708 is created in a note window 2706. This subnote 2708 is linked to the selected portion 2704. The user can enter comments pertaining to the selected portion 2704 in the subnote 2708 (that is, the user can annotate the selected portion 2704).

Group Annotations

A group annotation is not attached to a data object or a portion of a data object. The scope of a group annotation is the group that was selected or active at the time that it was created, and the group annotation is linked to the group. A group annotation is visible only when its associated group is selected or active. Preferably, any user with access to a group has access to the group annotations associated with the group.

Data Object Type Annotations

Data object type annotations are similar to group annotations. Data object type annotations are not attached to data objects. Instead, data object type annotations are linked to data object types. The scope of data object type annotations is the associated data object type that was specified when the annotation was created. Data object type annotations are only accessible when their associated data object types are selected or active. Like group annotations, access to data object type annotations is generally open to any user who has access to the associated data object type.

Case Annotations

A case is a data structure that includes a plurality of data objects. Such data objects may be in one or more groups, one or more data object types, and/or one or more other cases. A case may span multiple groups, data object types, other cases, etc. That is, a case can be set to include a given set of groups, data object types, cases, etc. A case is a type of context.

A case annotation is attached or linked to a portion of a data object contained in a case. The scope of the case annotation is the case that was active or selected when the case annotation was created. Case annotations are only visible if the associated case is selected or active. Access to case annotations is preferably under the control of the case owner. In an embodiment, a document that is being annotated is added to the active case automatically.

Enterprise Annotations

Enterprise annotations are attached and linked to data objects. The scope of an enterprise application is the data object to which it is linked. Accordingly, an enterprise annotation is accessible whenever the data object is accessible. Therefore, whenever a data object is contained in the active context, the enterprise annotations linked to that data object are accessible. In an embodiment, access to enterprise annotations is limited to trusted individuals, such as administrators or administrator designees, given the wide scope of enterprise annotation

In an embodiment, the ability to create an enterprise annotation is granted to the user by the system administrator. The administrator would specify the types of data objects that can be annotated by a user (e.g., NDAs, patents, license agreements, and so forth).

In an embodiment, enterprise annotations are only visible to a given user if the creator of the enterprise annotations exported his or her annotations to the given user, and the given user subscribed to the enterprise annotations of the creator. For example, if user 1 created an enterprising annotation 1 linked to data object 1, then user 2 when viewing data object 1 would only have access to enterprise annotation 1 if user 1 exported his enterprise annotations to user 2, and user 2 subscribed to the enterprise annotations of user 1. This exporting/subscription feature of the invention operates as both a security feature (enabling a creator to limit access to only those people to which his enterprise annotations have been exported to), and a filtering tool (enabling a user to display only those enterprise annotations generated by creators to which the user has subscribed).

User Interface and Note Processing

As noted above, the annotations pane contained in the browser or console window lists the currently active annotations. An annotation is active if its context is active. For example, a document annotation is active if it is linked to a data object contained in the active group (*i.e.*, the group selected in the group pane of the browser). A case annotation is active if its associated case is active (that is, its associated case is selected). A group annotation is active if its associated group is active (that is, its associated group is selected in the group pane of the browser). A type annotation is active if its associated type is active (that is, its associated type is selected in the type pane of the context browser). An enterprise annotation is active if the data object to which it is linked is contained in the active context (that is, it is contained in the active group or the active data object type).

Embodiments of the invention support a variety of formats for the annotation pane. Such formats are presented in the figures. For example, FIG. 31 illustrates an annotation pane 3102 having a tab for each annotation type. Specifically, there is a tab 3106 for document annotations, a tab 3108 for case annotations, a tab 3110 for group annotations, a tab 3112 for type annotations, and a tab 3114 for enterprise annotations. To view the active annotations of any given type, the user selects the tab associated with the type. For example, to view the currently active enterprise annotations, the user selects the enterprise tab 3114.

In embodiments of the invention, different types of annotations are denoted by different types of attributes. For example, different annotations may be displayed using different display attributes, such as font, color, size, underlining, icons, etc.

It is noted that the annotation type supported by the invention represent additional types of contexts. For example, enterprise annotations represent a type

of context that include either the active enterprise annotations, and/or the data objects linked to the active enterprise annotations.

FIG. 32 illustrates a flowchart 3202 illustrating the operation of the invention when creating an annotation that linked to a data object (that is, when creating a document annotation, a case annotation, or an enterprise annotation).

In step 3206, the user selects an annotation mode. This can be done, for example, by selecting an annotation pen 3001 (see, for example, pens 3001 in FIG. 30). It is noted that pens can be assigned to different types of annotation types such that the type of annotation to be created depends on the pen that is selected. Alternatively, this can be done by selecting an appropriate menu command, such as a drop down menu or a menu that it appears upon pressing the right mouse button.

In step 3208, the user selects a portion of a data object using the selected annotation pen, if any.

In step 3210, the user selects an annotation type (*i.e.*, a document, case, or enterprise annotation). Its noted that step 3210 is performed only when the user has not previously selected an annotation type. For example, as noted above, the user may have previously selected an annotation type when he selected the annotation pen 3001.

In step 3212, the system creates a new annotation of the type selected in step 3210, and links the annotation to the portion of the data object selected in 3208. Further in step 3212, the user enters comments or notes to annotate the selected data object portion.

In step 3214, information pertaining to the new annotation, including the user entered comments, the linking information, etc., is stored in an annotation database.

FIG. 33 illustrates a flowchart 3302 that represents the operation of the invention when creating a group or a data object type annotation (that is, when creating an annotation that is not linked to a data object).

In step 3306, the user selects a group or a data object type. The selected group or data object type becomes the active group or data object type.

In step 3308, the user selects a group or type annotation mode. This can be done, for example, by selecting an appropriate menu command, or by
5 selecting an appropriate annotation pen 3001.

In step 3310, the system creates a new annotation of the selected type. The new annotation is linked to the active or selected group or data object type. Also in step 3310, the user enters notes or comments to thereby annotate the
10 active group or data object type.

In step 3312, information pertaining to the annotation, such as the user's comments, the linking information, etc., are stored in an annotation database.

Form-based Annotations

As noted elsewhere, the present invention supports annotation of data objects. For example, FIG. 78A illustrates an example window 7802 wherein a
15 data object 7803 is being displayed. The present invention enables annotation of any portion of the data object 7803.

The window 7802 includes a plurality of annotation pens 7804. The user places the system in an annotation mode by selecting one of the annotation pens 7804 (the invention includes other ways of placing the system in the annotation
20 mode, such as selecting an appropriate command from a menu or dialog). The user utilizes the selected annotation pen to select a portion of the data object 7803. For example, FIG. 78B illustrates a selected portion 7806 of the data object 7803 that has been selected using the selected annotation pen. Upon selecting the selected portion 7806, the system automatically creates a new
25 annotation, which is illustrated as note segment 7810 in note 7808. The note segment 7810 is automatically linked to the selected portion 7806.

Preferably, the cursor is automatically positioned in an annotation field 7814 of the note segment 7810 so the user can record his observations. When finished, the user presses the save button 7850. Upon pressing the save button 7850, information pertaining to the annotation, such as the user's comments in the annotation window 7814, linking information, location information, the name of the annotation, whether the annotation includes privileged information, etc., are stored in an appropriate data format. Preferably, in a single user system, such annotation information is stored on the local computer. In a multi-user system, the annotation information is preferably stored in a database in such a way that the information can be securely shared with other users.

In the embodiment being described, the system does not provide any guidelines or place any restrictions on the scope and content of the comments. Accordingly, comments entered by the user may be of no value to himself or others. In some cases, the user's comments may be inappropriate given the circumstances under which the comments were solicited.

The user's comments comprise free form text which is entered in the annotation field 7814. In the embodiment being described, the comments are stored in an unstructured data format that is appropriate for storing free text, such as but not limited to a flat file database or a binary large object (BLOB) in a relational database. This data format is not structured in any way and is therefore not appropriate for storing in a standard relational database format.

According to an embodiment, the invention supports form-based annotations. Form-based annotations extend the operation of the annotation system described above to allow an expert user to create a specific form that is directed and limited to the information that one wishes the user to provide. Herein, for reference purposes, the person who creates a form is referred to as the form creator and the person who uses the form in the process of making an annotation is the annotation creator. Preferably, the form-based annotation system operates in a secure, multi-user, client/server system.

FIG. 79 illustrates an example annotation form, which is called the screening form 7902. According to the invention, annotation forms include any well known input widgets, such as radio buttons (choose one from many items), check boxes (which represents, for example, yes/no or true/false), text input fields (that may include edit checks), etc. For example, the annotation form 7902 includes radio button fields 7906, 7908, 7912, a structured date field 7910, and a name field 7904 which includes an edit check 7914 to ensure that proper data is entered.

According to the invention, data entered into one or more fields of an annotation form are stored in relational database fields. The number of annotation form fields that are stored in relational database fields vary from form to form, depending on the specific implementation of the form creator.

In the example of the annotation form 7902 of FIG. 79, information entered into the name annotation form field 7904 is stored in a name field 7926 of a relational database table 7922. Information from the radio button fields 7906, 7908, and 7912 are stored in fields 7928, 7930, and 7936, respectively, of the relational database table 7922. Date information entered into the when annotation form field 7910 is stored in a date field 7932 of the relational database table 7922.

FIG. 80 illustrates another example annotation form, called the comments form 8002. In the example embodiment of FIG. 80, information entered in the comments form 8002 is stored in either a structured table 8016 or a free form table 8018. The structured table 8016 and the free form table 8018 collectively comprise a comments form database 8014. In particular, name information entered into a name annotation field 8004 is stored in a name field 8022 of the structured table 8016. Information from radio button fields 8006 and 8008 are stored in a product 1 field 8024 and a product 2 field 8026, respectively, of the structured table 8016. Free text information entered into a comments on claim

1 field 8010 and a comments on claim 2 field 8012 are stored in rows of the free form table 8018.

5 It should be understood that the particular database table implementations of the screening form 7902 and the comments form 8002 shown in FIGS. 79 and 80 are provided for purposes of illustration only, and not limitation. Other database and table implementations will be apparent to persons skilled in the relevant arts based on the teachings contained herein. Also, the particular configurations and arrangements of the screening form 7902 and the comments form 8002 are provided for illustrative purposes only, and are not limiting. 10 Other forms will be apparent to persons skilled in the relevant arts based on the teachings contained herein.

FIG. 82 illustrates a flow chart 8202 that represents the operation of the form-based annotation system according to an embodiment of the invention.

15 In step 8204, a form creator creates a form, such as the screening form 7902 or the comments form 8002. The form creator is any user who has been designated as a person capable of creating and editing forms.

20 In step 8206, one or more relational database tables are created for the form, if necessary. Information entered into the form created in step 8204 will be stored in these database tables. These database tables may be created by the form creator or by some other user, such as but not limited to a system administrator.

25 In step 8208, the form creator selects one or more relational database tables for storing the information that is to be entered into the form. The selected database tables may include, for example, one or more of the tables created in step 8206, and/or other database tables created at other times.

In step 8210, the form creator assigns one or more fields (columns) from the selected relational database tables to each of the fields of the form. Data entered into the fields of the form are stored in the selected database table fields.

5 In step 8212, the form creator specifies and defines any additional features of the form. For example, the form creator can specify processing associated with any of the fields of the form. Considering FIG. 79, for example, the name field 7904 involves processing 7914 to check name information against a company directory to ensure that the name information was entered correctly into the name field 7904. The screening form 7902 also includes processing 7916 associated with the when field 7910 that involves confirming that the date entered into the when field 7910 is a valid date. The when field 7910 includes further processing 7920 associated with comparing the date entered into the when field 7910 with a projected IDS (information disclosure statement) filing date 7918 to identify issues associated with submitting references to the U.S. Patent Office. For example, if the projected IDS filing date 7918 is greater than 3 months after the date entered into the when field 7910, then a flag is set in database field 7934. The flag indicates that there may be an issue relating to whether or not the reference can be cited with the U.S. Patent Office in satisfaction of the duty of disclosure for a pending U.S. patent application. Other form features that can be specified in step 8212 is further described below.

15 After step 8212 is completed, the design of the form is complete. Steps 8214, 8216, and 8218 involve the manner in which an annotation creator uses a form.

20 In step 8214, the annotation creator selects a form. The form is tied to a specific pen (or some other annotation invocation mechanism, such as an option on a menu). The annotation creator selects the pen for the form, thereby informing the system which form to invoke, how to collect/store the data, etc.

25 In step 8216, the selected form is displayed. The annotation creator enters information into the form as specified by the fields of the form.

In step 8218, information entered into the fields of the form is stored in the associated relational database fields as defined by step 8210.

The operation of the form-based annotation system of the present invention is further described with reference to a flow chart 8302 shown in FIGS. 83A and 83B. Steps 8304, 8306, and 8308 of flow chart 8302 generally correspond to steps 8204, 8206, 8208, 8210, and 8212 of flowchart 8202 of FIG. 82. Step 8310 of flowchart 8302 generally corresponds to steps 8214, 8216, and 8218 of flowchart 8202 of FIG. 82.

In step 8304, preferably the system administrator authorizes a given user (the form creator) to create forms. In other words, the ability to create, modify, edit, and otherwise operate with forms is preferably a secured operation. This is because the creation of the form includes appropriate provisions for storing the data acquired for the use of the form in relational database(s) for backend statistical processing, as well as other processing.

In step 8306, the form creator logs into the system and creates a form to capture the data he is interested in capturing.

The invention supports all known input widgets, including radio buttons, text input fields, check boxes, etc. The form creator explicitly or implicitly associates input widgets with back end database tables. Even the meta content of the tables may be stored with the form allowing for the use of sophisticated data mining applications of the form and the data. It is also possible to associate programmatic operations with the form. Examples of such operations include edit checks on input fields, processing based on data entered into fields, invocation of other applications based on information entered into fields, etc.

In step 8308, the form creator associates the form with a pen (such as one of the annotation pens 7804 as shown in FIG. 78A) or other annotation mechanism (such as a pull down menu). This is part of a publishing process in which the form creator also specifies appropriate identifying characteristics of the form (such as a name of the form) and the storage locations of the data that is to be captured by form (that is, the relational database table fields in which the data entered into the form is stored).

5 The publishing process also allows the form creator to specify which user or groups of users should be allowed access to the form. Users are preferably identified by either their log in I.D. (identification) or by a role I.D. that specifies the role that they are assuming with respect to the operation of the system. In this way, the form creator can customize the operation and presentation of a given form-based on the user's role. For example, for a particular form, some fields are shown to marketing people, whereas other fields are shown to engineering people. Some fields may be shown to both marketing and engineering people. Further, some processing may be performed when the user has a marketing role, and other processing may be performed when the user has an engineering role. Some processing may be shown to both engineering and marketing people.

10 In an embodiment, the role based process described above allows a person to play multiple roles at the same time. Thus, a marketing/engineering role would see both marketing and engineering forms.

15 The form creator can also associate multiple forms with a single annotation mechanism. The form creator can further decide if they want the system to route the form to specific users and, if so, the manner of routing.

20 After completion of step 8308, design of the form is complete. In step 8310, the annotation creator uses the form to create an annotation. In particular, the annotation creator creates a new annotation by possibly using a pen, as described above. The system invokes the form associated with the pen and presents it to the user. In a case that multiple forms have been associated with a single annotation mechanism, the system presents a choice of forms to the user for selection (e.g., through a pull down menu, or pick list). Alternatively, the system sequences through the forms associated with the annotation mechanism. The user enters the information called for by the form, and the system stores this information in the appropriate relational database locations (and potentially other

25

non-relational database locations) as specified when the form was being designed.

The form-based annotation system will now be further described by considering a number of examples.

5 *Medical Example*

As noted above, form-based annotations can be used to better ensure that specific and desired information is collected from the user. Consider a doctor who has asked a colleague to review an X-ray to determine if a specific medical condition is present. The colleague may not fully understand the request, and enter irrelevant information as an observation if they are using a text field for annotations, as with the "free-text" annotation system of FIGS. 78A and 78B. However, the doctor could create a form that asks specific medical questions. In this case, when their colleague makes an annotation, he would be prompted for the desired medical input.

15 *Legal Example*

A lawyer may wish to have an engineer comment on one or more claims of a patent, but would like to control the specific commentary that can be entered by the engineer. With free-text annotations, the engineer could inadvertently enter inappropriate information. With form-based annotations, the lawyer could limit the amount and kind of information that an engineer could enter.

For example, consider the scenario shown in FIG. 81. A novice user 8108 fills out a screening form 8106 (see FIG. 79), which is then stored in the screening form database 7922. The completed screening form 8106 is also automatically routed to the IP (intellectual property) attorney 8110 (this operation is programmed into the screening form). If, in completing the

screening form 8106, the user 8108 checked the "Do you believe we should further study this patent?" field, then the comments form (FIG. 80) is routed to the novice user 8108 and an expert user 8120 to solicit their comments (this operation is programmed into the screening form).

5 The comments form is completed by both the novice user 8108 and the expert user 8120. Their completed comments forms 8118, 8124 are stored in the comments form database 8014. Also, the novice user's completed comments form 8118 is routed to the expert user 8120 for review (this is programmed into the comments form). The expert user 8120 may revise the novice user's
10 completed comments form 8118. The revised comment form is indicated as 8122. The revised comment form 8122 is stored in the comments form database 8014.

 Similarly, the completed comments forms 8118 and 8124 are routed to the IP attorney 8110, who may enter his own comments 8112, which are stored
15 in the comments form database 8014 (this is programmed into the comments form).

Product Development Example

 There are times when one user, who is an expert in one domain, wants specific opinions from one or more users whose expertise lies in different, but
20 related, domains. In this case the first expert could annotate a specific portion of a data object. When another expert accesses this annotation the system will prompt them, through the use of a form-based annotation, for their opinion of the annotated data object. The system provides a wide variety of mechanisms to obtain feedback from expert users, as well as other selected users, including
25 routing and multiple routing slips and time-based expirations.

 Consider a product development example where a product marketing manager may want an engineer to provide a cost estimate on how much it would

cost to manufacture a patented product to help determine long-term product strategy. In this case, the marketing manager would associate appropriate edit-checks on the input widgets of the form to ensure that the engineers are entering in numeric information in a reasonable range. Because a decision is needed quickly the marketing manager routes it through the company email system to all of "ENGINEERING" specifying a time-limit of 7 days. If an engineer does not respond within 7 days the system automatically removes the request from their "work queue".

Note that an engineer may annotate the patent several times, thereby providing multiple cost estimates, one for each annotated portion of the patent that would, in the opinion of the engineer, result in a component or portion of the product. In this case, the system would operate in a manner similar to free-text annotations, automatically creating sub-forms contained within a larger "form" (much like creating sub-notes in the context of a note). The form-creator would have to decide on the operation of these forms during the creation process (e.g., the form-creator would either allow or reject the use of multiple sub-forms within a form).

Additional Product Development Example

Form-based annotations enable the collection and storage of observations of users who make annotations in a way that these observations can be used for reliable reporting and other statistical data mining operations.

Consider the product development example discussed above. After 7 days the product marketing manager could run a report that produces the weighted average of the cost estimates from the engineers. The system could perform this report by querying specific database columns, and extracting data from specific database columns. For example, the system can execute a search in a Department column of the relational database to identify all rows of the

database that correspond to annotations made by members of a given engineering department. The system can then extract data from the "Cost Estimate" column of the identified rows.

Accordingly, the form-based annotation system of the invention facilitates data mining, statistical processing, and other processing of the annotation data.

If, instead, a free text format was used, it would be necessary to parse the free-text to extract the desired information.

It is noted that the invention is not limited to only storing information in formally structured fields within a relational database. In fact, a form could contain a mixture of structured fields (such as dates, numbers, or Boolean) and unstructured text fields, as discussed above.

Medical Teaching Example

With free text annotations, for example, the user selects a portion of a data object and then describes why the selected portion is relevant. Sometimes this is exactly the opposite order of what is desired. Specifically, sometimes an expert user wants a different (and perhaps more novice) user to annotate a portion of a data object in response to what the expert user created.

For example, a teacher could create the contents of an annotation and ask students to "find the spot where the annotation is true". For example, a teacher could prepare a series of x-rays of difficult to diagnose fractures. The assignment could be: "Identify the location of the hairline fracture". Students would then annotate the data object where they thought the fracture was using a form-based annotation, possibly adding commentary on their selection in a new annotation window (or in the original). The teacher could then compare the work of the student to a known standard.

Searching

The invention supports a variety of types of data objects. For example, the invention supports patent related data objects, corporate documents (which, in some embodiments, constitute non-patent related data objects), etc. Such data objects are preferably stored in system databases, such as the IPAM databases 7614 and/or local databases 7612 (see FIG. 76). The invention also supports data objects that are not stored in system databases. For example, the invention supports data objects that are available from third parties. Such third parties include, but are not limited to, on-line data providers, such as LEXIS-NEXIS, Dialog, Westlaw, Derwent, etc. Third parties also include, but are not limited to, public or government databases, such as those associated with the SEC, FCC, FDA, Department of Motor Vehicles, Social Security, etc. With regard to such third party providers, the invention supports establishing a link with the third parties, and enabling users of the invention to interact with the third parties via the link. Such third party providers include well known interfaces or APIs (application programming interfaces). In embodiments, the invention interacts with such third parties via their respective interfaces or APIs.

The invention includes functionality to enable users to conduct complex searches through the data objects that it supports. The invention provides detailed graphical user interfaces (GUI) to enable users to define their searches. FIG. 34 illustrates an example search GUI 3402. In the example of FIG. 34, the search GUI 3402 includes a variety of tabs associated with the data object types supported by the invention.

For example, the invention includes a patent documents tab 3404 for specifying a search among the patents stored in the system databases. The search GUI 3402 also includes a corporate documents tab 3406 for specifying a search among the corporate documents stored in the system (see FIG. 35). The search

GUI 3402 also includes an all documents tab 3408 for specifying a search among all data objects stored in or supported by the system (see FIG. 36).

The search GUI 3402 further includes tabs for third party information providers supported by the system. For example, and without limitation, a LEXIS-NEXIS tab 3410 in the search GUI 3402 is selected to define a search to be performed by the LEXIS-NEXIS system. In an embodiment, upon providing such search criteria, a link is established with the third party information provider wherein the third party is instructed to perform the specified search. Search results are then returned to the system (i.e., the enterprise server 7604, the network client 7608, and/or the web client 7606). Such information is then treated as any other system information.

Further details pertaining to the searching capabilities of the invention are described in U.S. Patent Application "System, Method, and Computer Program Product for Patent-Centric and Group-Oriented Data Processing," Ser. No. 08/867,392, incorporated herein by reference in its entirety.

Data Presenting and Processing Using Visualization Technologies, Such as Hyperbolic Trees

The invention supports the use of a variety of visualization technologies to present information to users. Such technologies include, but are not limited to, hyperbolic trees. Other technologies useful for presenting data to users are discussed in sections below.

A hyperbolic tree or a hyperbolic browser is a well known "focus plus context" technique for visualizing and manipulating data hierarchies, such as trees. A hyperbolic browser assigns for display a portion of a tree while still embedding in it the context of the entire tree. The essence of this scheme is to lay out the tree in a uniform way on a hyperbolic plane and map this plane onto a display region, such as but not limited to a circular display region (other

display geometries are also possible). Thus, hyperbolic trees support a smooth blending between focus and content, as well as continuous redirection of the focus.

5 An example hyperbolic tree 3702 is illustrated in FIG. 37. The hyperbolic tree 3702 includes a root node 3704 and a plurality of additional nodes connected either directly or indirectly to the root node 3704. Additional portions of the hyperbolic tree 3702 can be expanded and displayed in the window 3712 (that is, focus can be redirected to those portions) by traversing to the desired portions in the hyperbolic tree 3702. An operator traverses the
10 hyperbolic tree 3702 using a pointing device, such as a mouse.

According to the invention, the hierarchical information represented in hyperbolic trees may come from any source, including but not limited to patent offices, corporate entities, public records, individual users, etc. In an embodiment, the hierarchical information is patent related, but the invention is
15 not limited to this embodiment.

The invention provides a variety of functions to enable users to customize the display of hyperbolic trees. For example, embodiments of the invention store state information in each node of the hyperbolic tree.

FIG. 42B illustrates example node state information 4202 which is stored
20 in each node of the hyperbolic tree. The node state information 4202 includes the label that is displayed in the associated node. For example, the label of node 3706 in FIG. 37 is "claim 271."

The node state information 4202 also includes display attribute information that indicates the manner in which the node is to be displayed in the window 3712. Such display attributes include any attributes for displaying
25 information on a computer screen, such as font, color, bold, italics, underlining, visible/invisible, etc. The display attributes also include user defined and custom attributes, which are entered by a user. The user defined attributes are useful to

enable users to specify the display of nodes to meet their specific applications.

In some embodiments, the display attributes of a node are automatically set based on the information corresponding to the node. For example, when a hyperbolic tree is used to display a patent claim tree, the display attributes of nodes may depend on whether the nodes correspond to a patent or a claim, an independent claim or a dependent claim, a disclaimed claim, a valid claim, an invalid claim, an infringed claim, a non-infringed claim, a claim that has been annotated, etc.

In some embodiments, the display attributes may be set according to information entered into the user defined field. For example, if the user enters that a given claim is invalid, then the font and color display attributes may be set to preselected values corresponding to invalid claims. Therefore, the user defined field(s) and the display attributes portion of the node state information 4202 may specify actual display attributes (such as crosshatching), or state information that the system uses to automatically set the display attributes.

The node state information 4202 also includes link information that establishes a link between the node and associated information in the system databases (i.e., the IPAM databases 7614 and/or the local databases 7612 shown in FIG. 76). In the case of a claim tree 3702 in FIG. 37, the link information establishes a link between the node and the text of the claim associated with the node. Alternatively, such claim text can be stored in the node itself as part of the node state information 4202. In other embodiments, a portion of the claim text (such as the preamble or a portion thereof) is stored in the node as part of the node state information 4202, and the link information in the node state information 4202 establishes a link to all of or the remaining portion of the claim text.

The invention supports functions to edit the appearance of a hyperbolic tree. For example, the invention allows a user to prune a tree by removing nodes

or branches. In embodiments of the invention, such pruning is accomplished by deleting the identified nodes and/or branches from the tree. In other embodiments of the invention, such pruning is accomplished by appropriate setting of the display attributes of the selected nodes and/or branches. Specifically, nodes corresponding to portions of the tree which a user has selected to delete are set to invisible or transparent.

Specific embodiments of hyperbolic trees supported by the invention are described below. It is noted that these embodiments are provided for illustrative purposes only, and are not limiting. Additional applications of hyperbolic trees will be apparent to persons skilled in the relevant art(s) based on the teachings contained herein.

Patent Claim Trees

As noted above, the invention includes a patent claim tree module that supports patent claim trees.

The patent claim tree module generates a graphically displayed claim tree from a collection of claims. The patent claim tree module works both as a standalone product, and as an add-on to IPAM.

The patent claim tree module includes a mechanism to elicit help from the user when it fails to understand enough about a claim to perform its function. Whenever the patent claim tree module experiences ambiguity or failure in the processing of a claim, it will produce a log entry containing the text of the claim, the claim properties inferred by the patent claim tree module, and any corrections brought about by the user.

The patent claim tree module preferably uses the graphical display of the tree to convey visually the properties of individual claims. These properties include, but are not limited to:

- * Root/Independent/ Dependent

* Claim type (Method, Composition of matter, Machine,...)

Other properties are described herein.

In some embodiments, the style of claims supported by the patent claim tree module is limited to the one prevalent in standard utility patents. As a result, the patent claim tree module covers claim sections contained in any document whose style is substantially identical to that of utility patents (e.g. Reissues, Statutory inventions). Alternative embodiments support other claim styles that are different (e.g. Design patents, Plant patents).

FIG. 87 illustrates an example patent claim tree module 8706, which is also herein called a claims processor. The patent claim tree module 8706 is composed of three main stages: A Preprocessor, an Aggregator, and a claim parser.

The responsibilities of the Preprocessor include:

- * Remove any leading or trailing text
- * Divide the text into a collection of claims
- * Turn the input character stream into a token stream that only contains the words that may be of use to the rest of the process.

The Aggregator further turns sets of tokens following certain patterns into single tokens.

The claim parser uses the token string corresponding to each claim to set the properties of that claim as well as its dependency relationships to other claims. Its final product is preferably a directed a-cyclic graph representing the claims of the collection, their properties, and their dependencies.

An example claim tree 3702 is shown in FIG. 37. In the claim tree 3702, the root node 3704 represents the patent or the patent application. In the example of FIG. 37, the claim tree 3702 corresponds to a patent application having Attorney Docket No. 1531_0010006.

The nodes directly connected to the root node 3704, such as the nodes corresponding to claims 181, 208, 235, 271, 272, 273, and 274, represent the

independent claims in the patent or patent application. The nodes connected to these independent claim nodes represent the dependent claims of the respective independent claims. The links in the patent claim tree 3702 represent the dependencies between claims.

5 The invention supports a number of functions to facilitate the display and analysis of claims. For example, when the cursor is positioned proximate to or over a claim node, a portion of the claim is displayed (in other embodiments, the entire claim is displayed). In FIG. 37, for example, the cursor 3708 is displayed over node 3706 corresponding to claim 271. According to this embodiment of
10 the invention, a portion of claim 271 is retrieved and displayed in window 3710. This portion of the claim can be retrieved from the node state information 4202, if available, or can be retrieved from a system database (such as the IPAM databases 7614 and/or the local databases 7612) by access to the link information in the node state information 4202.

15 The invention also supports a "display claim branch" function. As is well known, a dependent claim is construed to include the base independent claim and any intervening claims. When the user positions the cursor over a dependent claim and selects the "display claim branch" command, the system retrieves the text of the selected claim, its base independent claim, and any intervening claims
20 between the selected claim and the base independent claim. Such retrieval can be accomplished, for example, by following the links in the claim tree 3702 from the selected claim to the associated independent claim. The invention then displays the retrieved claim language in the order of dependency. In an embodiment, the retrieved claim text is concatenated in order of the independent
25 base claim, the intervening claims, and finally the selected dependent claim.

 For example, assume that the user elected to perform a "display claim branch" function on claim 270 shown in FIG. 37 as being dependent on claim 235. The text of claim 270 is illustrated in window 4002 of FIG. 40. The text of claim 235 is illustrated in window 3902 of FIG. 39. Upon issuing the "display

claim branch" command on claim 270, the invention retrieves the text of claims 270 and 235. The invention then concatenates the retrieved text in order of dependency. The concatenated text is then displayed in a window, for example, window 4102 of FIG. 41. Note that the information displayed in window 4102 has a portion 4104 corresponding to claim 235, and a portion 4106 corresponding to claim 270.

Other display formats for presenting and displaying the retrieved claim text will be apparent to persons skilled in the relevant arts based on the discussion contained herein.

Claim trees are useful for a number of applications, including but not limited to:

- Automatically calculating claims fees for adding independent and dependent claims to a pending application or at the time a case is allowed.
- Separating out "method" and "process" claims from preamble search criteria.
- Identifying "system" claims from preamble search criteria.
- Identifying "compound" and "chemical structure" claims from preamble search criteria.
- Identifying "Jepson" type claims by searching for appropriate preamble language.
- Showing "Once Amended", "Twice Amended" and similar distinctions in claims submitted by adding these terms to the search criteria immediately following the claim number.
- Graphically showing claim dependency for ease of analysis.
- Graphically showing claim dependency to determine claim scope and coverage.
- Generating claim charts for presentations.

The claim tree functionality of the present invention supports performance of these tasks.

FIG. 38 includes a flowchart 3802 that represents the operation of the invention when generating and processing claim trees.

In step 3806, a claim is selected for processing. As noted above, the claim can be from a patent or a patent application, as long as the patent or patent application is stored in system databases.

In step 3808, the invention determines whether the selected claim is an independent or dependent claim. This is done by parsing the claim text to search for keywords that indicate whether the claim is independent or dependent. For example, the claim text may be parsed for the word "claim" in the preamble, which would indicate that the selected claim is a dependent claim.

If the claim is determined to be a dependent claim, then in step 3810 the claim text is parsed to identify the claim or claims on which it depends.

Accordingly, steps 3808 and 3810 correspond to parsing the claim language to identify claim properties and information used to generate the claim tree. Embodiments of the invention operate to recognize key words, phrases, and/or patterns in claims to identify such claim properties and information. The invention is flexible in this regard.

The need for flexibility in this area is motivated by the fact that, even though the language and structure of claims is constrained, claims are still expressed in natural language and present substantial variations in the terms and linguistic patterns used. This requires the patent claim tree module to be easily extensible to recognize new words and linguistic patterns that denote properties supported by the patent claim tree module.

Therefore, embodiments of the patent claim tree module include a configuration file. The configuration file includes information that:

* enable specification of words and phrases that denote properties identified and supported by the patent claim tree module (for example, one of these items will be the claim type of Process, which can be denoted, for example, by either the word "method" or the word "process"); and

* enable specification of words and phrases the patent claim tree module must recognize to correctly segment the text into individual claims, and the claims into sections. Two such phrases are, for example, "as claimed in claim," "in accordance with claim", or "of claim," which introduce one or more claims a claim depends on.

The configuration file can be modified.

In particular, the invention allows the addition of new textual patterns.

Also, in some cases, the user is ask to specify a textual pattern that may help the patent claim tree module deal with a given situation in the future.

Steps 3806, 3808, and 3810 are performed for each claim in the patent and/or application, as indicated by control flow line 3811. Based on the information obtained from processing steps 3806, 3808, and 3810, a claim dependency graph is generated in step 3812.

In step 3814, the claim dependency graph is mapped to a hyperbolic tree or another presentation tool.

In step 3816, the hyperbolic tree is displayed. The display of the hyperbolic tree utilizes the attributes stored in the node state information 4202 of each node. As noted above, such attributes may have been automatically established and processed based on information stored in the system, or may be user defined. The user may enter user defined attribute information by selecting an appropriate menu command, as shown in menu 4202 of FIG. 42A.

In step 3818, if the mouse is positioned proximate to or over a node, at least a portion of the text of the claim corresponding to the node is retrieved and displayed. This capability of the invention is further described above.

In step 3820, if the user selects a dependent claim and selects the "display claim branch" command (see menu 4202 in FIG. 42A), then the "display claim branch" function for the selected claim is performed. The "display claim branch" function is described above.

A user can elect to display the full text and/or image of any claim in the tree. This is done by positioning the cursor proximate to or over the node corresponding to the claim, and then selecting the "display claim" command from menu 4202.

5 The invention supports other processing of patent claims. For example, the invention supports automatic processing to compare a claim to an arbitrary text description.

10 Such processing can be used to analyze a claim for infringement purposes. In this application, the claim is compared to a description of a device or process that is accused of infringing the claim. Possible infringement is indicated if there is a close similarity between the claim and the device/process description.

15 Such processing can also be used to analyze a claim for patentability/validity purposes. In this application, the claim is compared to a description of one or more publications and/or events that might possibly render the claim unpatentable. Possible unpatentability/invalidity is indicated if there is a close similarity between the claim and the publications/events description.

20 Such processing can also be used to analyze a claim to determine if sufficient support for the claim is contained in the patent specification. In this application, the claim is compared to the specification. Probable support for the claim is indicated if there is a close similarity between the claim and the specification.

25 In an embodiment, the automatic claim comparison process of the invention is performed by normalizing the claim language of the claim through an implementation dependent linguistic process. The description to which the claim is to be compared is then normalized through a corresponding (sometimes identical or similar) process. The normalized texts are then compared to identify similarities and differences. A conclusion is then reached based on the context

of the analysis (i.e., whether it is for purposes of infringement, patentability, validity, support, etc.)

An example of such processing is described in U.S. Patent No. 5,754,840, which is herein incorporated by reference in its entirety.

5 As noted above, the patent claim tree module is capable of working in a stand-alone mode. In this mode, an example use scenario is as follows:

1. The user invokes the patent claim tree module.
2. The patent claim tree module comes up. It includes a main input/output window.
- 10 3. The user pastes a claim collection into the patent claim tree module's main input window.
4. The user configures the patent claim tree module by adjusting any available options.
5. The user press the tree generation button.
- 15 6. The patent claim tree module produces a graphical representation of the claims tree within a special purpose window. Errors and warnings are output to the log window.

The patent claim tree module can also work with IPAM. In this mode, an example use scenario is as follows:

- 20 1. The user selects a document within IPAM.
2. The user applies the "view claim tree" command to it (e.g. available inside the right-click pop-up).
3. The command routine (within IPAM) opens the document, extracts the claim section, and invokes the patent claim tree module. The claim section is
- 25 passed to the patent claim tree module which comes up and pastes in its input/output window.
4. The user configures the patent claim tree module by adjusting any available options.
5. The user press the tree generation button.

6. The patent claim tree module produces a graphical representation of the claims tree within a special purpose window. Errors and warnings are output to the log window.

These use scenarios are provided for purposes of illustration only. The invention is not limited to these use scenarios. Other use scenarios will be apparent to persons skilled in the relevant art(s) based on the herein teachings.

Patent Citation Trees (patent and non-patent citations)

The invention supports patent citation trees. An example citation tree 4302 is shown in FIG. 43. In the patent citation tree 4302, each node represents a patent, and each link represents a citation. When performing a forward patent citation function, the links going from the root node toward the children nodes or leaves represent the directed association "is cited in," as in "parent node is cited in child node." When performing a backward citation function, the links going from the root node toward the leaves represent the directed association "cites," as in "parent node cites child node."

The invention supports displaying a variety of labels in the nodes of the tree. The labels may be any information pertaining to the data objects that they represent. For example, in the case where the data object is a patent, the labels may be any bibliographic information of the patent, including but not limited to the patent number, inventors, assignee, claim language (or excerpt), specification (or excerpt), drawing information such as an image of a figure, class/subclass, patent examiner, law firm, etc. The label that is displayed is user selectable.

Both patents and non-patent documents and data objects may be cited in any given patent. According to the present invention, both patents and non-patent documents/data objects are represented in the patent citation tree 4302. For example, assume that the patent citation tree 4302 is a backwards patent citation. According to this example, patent B and patent C represented by nodes

4306 and 4308, respectively, are cited in patent A represented by node 4304. Non-patent documents or data objects C and D, represented by nodes 4310 and 4312, are also cited in patent A.

5 According to the invention, information pertaining to the data objects in nodes of a hyperbolic tree, whether a patent citation tree or otherwise, can be retrieved and displayed by selecting the node. Such functionality is accomplished by, for example, reference to the link information in the node state information 4202, or by reference to appropriate database tables.

10 The display attributes of nodes can be set according to information pertaining to the data objects that they represent. In the case where the data objects are patents, for example, the display attributes of nodes can be set according to any bibliographic information of the patent, including but not limited to the patent number, inventors, assignee, claim language (or excerpt), specification (or excerpt), drawing information such as an image of a figure,
15 class/subclass, patent examiner, law firm, etc. For example, the user can indicate that patents assigned to company X be color coded blue, whereas patents assigned to company Y be color coded red. In response to this input, the system automatically color codes nodes based on their assignee information.

Alternatively, the user can manually set the display attributes of nodes.

20 It is noted that this aspect of the invention has been described in the context of patent citation trees. However, the invention is applicable to data objects of any type that cite or are cited by other data objects. Accordingly, more generally, the invention supports data object citation trees.

25 According to embodiments of the invention, citations in citation trees are not limited to data object references that appear in any particular location of the data object(s) in question. For example, a backward citation tree for a U.S. patent is not limited to citations that appear on the front page of the U.S. patent. Also, a citation tree for a technical article is not limited to citations contained in the

bibliography of the article. Instead, citation trees can include citations to data objects that can be referenced anywhere in the data object(s) in question.

Preferably, in the case of patents, the invention includes data/text processing techniques for culling/identifying citation references to patents and non-patent documents that are contained within the specification but not listed on the first page bibliographic section. Parsing techniques to identify data in data objects are well known, and any can be used to implement this aspect of the invention. Such references can be included in patent citation trees (they could be displayed using different display attributes to denote their existence in the body of the specification, or can be displayed in different trees).

Example Databases

Databases for implementing the patent citation function of the present invention are illustrated in FIGS. 44 and 45. FIG. 44 illustrates a patent ref table 4402 that stores information on patents that were cited during the prosecution of a given patent. The patent ref table 4402 includes a record for each patent that was cited during the prosecution of a given patent application. Each record of the patent ref table 4402 includes a document_ID that identifies a base patent, and a refpatentno that identifies a reference patent (i.e., a patent that is cited in the base patent).

FIG. 45 illustrates a non-patent ref table 4404 that stores information on non-patent data objects that were cited during the prosecution of a given patent. The non-patent ref table 4404 includes a record for each non-patent data object that was cited during the prosecution of a given patent application. Each record of the non-patent ref table 4404 includes a document_ID that identifies a base patent, and a RefNo that identifies a non-patent data object that was cited in the base patent.

Information pertaining to other databases used by the invention is contained in U.S. patent applications "System, Method, and Computer Program Product for Patent-Centric and Group-Oriented Data Processing," Ser. No. 08/867,392, and "System, Method and Computer Program Product for Managing and Analyzing Intellectual Property (IP) Related Transactions," Ser. No. 09/138,368, incorporated herein by reference in their entirety.

The operation of the patent ref table 4402 and the non-patent ref table 4404 is illustrated in FIGS. 46 and 47. FIG. 46 presents an example patent ref table 4602 corresponding to the patent citation tree 4302 shown in FIG. 43. The patent ref table 4602 indicates that patents B and C are cited in patent A and patent D is cited in patent B.

FIG. 47 illustrates an example non-patent ref table 4702 that also corresponds to the patent citation tree 4302 in FIG. 43. The non-patent ref table 4702 indicates that non-patent data objects C and D are cited in patent A, non-patent data object A is cited in patent B, and non-patent data object D is cited in patent C.

Additional details regarding patent citation trees according to the invention are discussed in U.S. patent applications "System, Method, and Computer Program Product for Patent-Centric and Group-Oriented Data Processing," Ser. No. 08/867,392, and "Using Hyperbolic Trees to Visualize Data Generated by Patent-Centric and Group-Oriented Data Processing," Ser. No. 08/921,369, incorporated herein by reference in their entirety.

Data Object Family Trees

The present invention supports data object families. A data object family comprises a collection of data objects that are somehow related, and information pertaining to the relationships between the data objects in the family.

There are many types of families and family relationships. For example, the patent citation tree described above is an example data object family where the family includes a base patent and patents and non-patent data objects that are cited in or that are cited by the base patent.

5 According to the invention, data object families are contexts for organization, visualization, analysis, and other processing, as described above.

The invention supports the creation, generation, visualization, manipulation, and other processing of data object families. Data object families may be system defined or user defined.

10 Data object families are further described in the following sections.

Examples of Data Object Families

A number of the data object families supported by the present invention are described below. The invention is not limited to these examples. Additional data object families will be apparent to persons skilled in the relevant arts based on the discussion contained herein.

15 FIG. 51 illustrates an example patent family chronology 5102, which is a type of a data object family. A patent family chronology 5102 includes patents, applications, and related data objects that stem from a base patent application. In the example of FIG. 51, the patent family chronology 5102 stems from application 1, which represents the base patent application. Application 1 spawned a continuation application (continuation 1), which spawned a continuation-in-part application (CIP 1). A trademark (trademark 1) is related to continuation 1, and a copyright (copyright 1) is related to CIP 1. Application 1 was developed from a lab notebook maintained by an inventor (lab notebook 1). Application 1 also spawned a patent cooperative treaty application (PCT 1), which was national filed in Japan (Japan 1) and Canada (Canada 1).

FIG. 53 illustrates an example assignee technology patent family 5302, which is another type of data object family. An assignee technology patent family includes patents that are assigned to the same corporate entity, and that are technologically related. Relationships between patents may reflect, for example, improvements to the technology.

In the example of FIG. 53, the assignee technology patent family 5302 includes a bike patent 1, a bike patent 2, and a bike patent 3. Bike patent 1 is the assignee's first and base patent on bikes. Bike patent 2 is an improvement upon bike patent 1, and bike patent 3 is an improvement upon bike patent 2.

According to the invention, data object families are displayed using any well known visualization technology, including but not limited to hyperbolic trees. Techniques for generating hyperbolic trees from hierarchical organized data (such as data object families) is discussed in U.S. patent application "Using Hyperbolic Trees to Visualize Data Generated by Patent-Centric and Group-Oriented Data Processing," Ser. No. 08/921,369, incorporated herein by reference in its entirety.

The data object families described above, as well as additional data object family examples supported by the present invention, are further described below.

Patent Related Families

The invention supports a plurality of patent-specific families. Examples of such families are described below.

Assignee Patent Family Chronology (APFC)

This family depicts the relationship between related issued patents and pending applications (and possibly related data objects) of the same assignee. This includes, but is not limited to, U.S. and corresponding foreign applications.

This includes the original application, divisions, continuations, CIPs, Reissues and Re-exams, Patent Extensions. The information to create these families could be found, for example, in the assignee's internal patent docket system.

5 This family would be used for understanding any company's particular family relating to a group of patents and applications that stem from the same original application. Such a family would show in a tree-type format the relationship between, for example all the patents and applications in a corporate entity's family. It helps a user see how a company has clustered improvement
10 patents around newly developed technology. It could be useful to product planning, product development, and the business and legal people in charge of managing the company's portfolio.

Assignee Technology Patent Family (ATPF)

15 This family shows the relationship between patents and applications of the same assignee in the same area of technology but not necessarily limited to an Assignee Patent Family Chronology. This family would be a broader grouping that would relate patents and applications of an assignee across an entire technology. The information to create these families could be found, for example, in the assignee's internal patent docket systems and keyword searching
20 on the assignee's patents and applications.

Technology Patent Family (TPF)

25 This family shows the relationship between patents in the same area of technology. This family is not limited by assignee. The information to create these families could be found, for example, by a combination of patent bibliographic and patent text searching. This family would be somewhat like a

forward citation tree but more limited based on the culling and grouping criteria used by the person creating the family.

Claim Trees

These families focus on the relationship between claims in terms of dependency and dominance-subservience among claims.

Patent Claim Tree (PCT) and Application Claim Tree (ACT)

These trees show the relationship between claims in a single issued patent or a pending application. This tree is preferably created by searching at least the first line of all claims in a given patent or application for an indication of dependency, and/or subject matter and scope. This is useful in patent prosecution, licensing and litigation scenarios in graphically representing the dependency relationship among claims in a given patent or application. Patent claim trees are further described above.

Assignee Patent Family Chronology Claim Tree (APFCCT)

This tree shows the relationship between claims in an Assignee Patent Family Chronology. This tree would show the independent-dependent claim relationships within a given patent or application and dominant and subservient or improvement claims in an Assignee Patent Family Chronology. The information to create these trees could be found, for example, in the assignee's internal patent docket systems and keyword searching on the assignee's patent

and application claim sets and the culling and grouping criteria used by the person creating the tree.

***Assignee Technology Patent Family Claim Tree
(ATPFCT)***

5

This family shows the relationship between claims in an Assignee Technology Patent Family. This would be similar to the Assignee Patent Family Chronology Claim Tree but would not be limited to a given Patent Family Chronology. This family covers the Assignee's entire related technology base.

10

This family shows the claim relationships of dependency and dominant-subservient claims within an Assignee's Technology Patent Family Chronology. The information to create these trees could be found, for example, in the assignee's internal patent docket systems and keyword searching on the assignee's patent and application claim sets and the culling and grouping criteria used by the person creating the tree.

15

Technology Patent Family Claim Tree (TPFCT)

This family shows the relationship between claims in all patents in a Technology Patent Family. This would be similar to the Assignee Technology Patent Family Claim Tree but would not be limited to a given Patent Family Chronology but would cover the entire related technology base for all assignees. The information to create these families could be found, for example, by a combination of patent bibliographic and patent text searching and the culling and grouping criteria used by the person creating the tree.

20

Patent-Trademark Relationship Trees

These families are trees that include both patent claims and trademarks used on the patented products covered by the particular patent claims. They are created in a similar way that the Patent Families and Trees would be created with the added dimension of including trademarks in the families and trees that correspond to the patents covering products sold using those trademarks.

Example Databases

Databases useful for implementing functions related to data object families are described in this section.

FIG. 48 illustrates a data object family table 4802 according to an embodiment of the present invention. Relationships between pairs of data objects are specified in the data object family table 4802. More specifically, the data object family table 4802 includes a record for each pair of data objects of interest. Each record of the data object family table 4802 includes a "from" field to identify a first data object, a "to" field to identify a second data object, and a relationship type field that stores information indicating the relationship between the two data objects.

FIGS. 50A and 50B illustrate an example relationship type table 5002 that specifies relationship types supported by the invention. The relationship types indicated in FIGS. 50A and 50B are provided for illustrative purposes, and are not limiting. In general, the relationship type table 5002 stores all relationships of interest to the user. Additional relationships will be apparent to persons skilled in the relevant art(s).

Each row of the relationship type table 5002 includes a relationship type name, the unique identifier for the relationship, and a grouping ID. The grouping ID is used to aggregate relationship types for a given data object family. For

example, grouping G1 generally corresponds to a patent family chronology. Grouping G3 generally corresponds to an assignee technology patent family.

The relationship type table 5002 includes relationships that are predefined (system defined), and relationships that are user defined. Accordingly, the invention allows users to define and/or modify relationships based on the needs of their particular applications. Additionally, the groupings contained in the relationship type table 5002 comprise both predefined groupings and user defined groupings. Accordingly, users can create and/or modify groupings based on their specific needs.

FIG. 49 illustrates an example data object family table 4902 corresponding to the patent citation tree 4302 shown in FIG. 43. The data object family table 4902 represents an alternative implementation of the patent citation tree 4302. Rows 4904-4916 all correspond to backward citation relationships, as indicated in the relationship type column. Specifically, rows 4904, 4906, 4908, and 4910 of the data object family table 4902 indicate that patent B, patent C, non-patent object C, and non-patent object D are referenced in patent A. Rows 4912 and 4914 of the data object family table indicate that patent D and non-patent data object A are referenced in patent B. Row 4916 of the data object family table 4902 indicates that non-patent data object D is referenced in patent C.

Rows 4918-4930 of the data object family table 4902 correspond to forward citation relationships, as indicated in the relationship type column. Specifically, row 4918 indicates that patent B is cited by patent A. Row 4920 indicates that patent D is cited by patent B. Row 4922 indicates that non-patent object A is cited by patent B. Row 4924 indicates that patent C is cited by patent A. Row 4926 indicates that non-patent data object D is cited by patent C. Row 4928 indicates that non-patent data object C is cited by patent A. Row 4930 indicates that non-patent data object D is cited by patent A.

Use of the data object family table is further illustrated in FIGS. 51 and 52. As discussed above, FIG. 51 illustrates an example patent family chronology 5102. FIG. 52 illustrates an example data object family table 5202 that corresponds to the patent family chronology 5102. Row 5204 of the data object family table 5202 indicates that lab notebook 1 represents invention disclosure materials used to generate application 1. Row 5206 indicates that continuation 1 is a continuation of application 1. Row 5208 indicates that CIP 1 is a continuation-in-part of continuation 1. Row 5210 indicates that copyright 1 is a copyright that is related to CIP 1. Row 5212 indicates that trademark 1 is a trademark that is related to continuation 1. Row 5214 indicates that PCT 1 is a PCT application from application 1. Row 5216 indicates that Japan 1 is a Japanese foreign national application from PCT 1. Row 5218 indicates that Canada 1 is a Canadian foreign national application from PCT 1.

In an embodiment, the patent family chronology 5102 of FIG. 51 is preferably generated by selecting grouping G1 from the relationship type table 5002. A data object family is then generated that includes the relationships within grouping G1. Further details regarding the generation of data object families is provided below.

Another example of the use of the data object family table is presented in FIGS. 53 and 54. As indicated above, FIG. 53 illustrates an example assignee technology patent family 5302. FIG. 54 illustrates a data object family table 5402 that corresponds to the assignee technology patent family 5302.

Row 5404 of the data object family table 5402 indicates that bike patent 2 is an improvement of bike patent 1. Row 5408 indicates that bike patent 3 is an improvement of bike patent 2. Additional information is contained in the data object family table 5402. For example, row 5406 indicates that bike patent 1 is technically related to bike patent 2, and is a base patent. Row 5410 indicates that the claims of bike patent 2 are subservient to the claims of bike patent 1. Row 5412 indicates that the claims of bike patent 2 are an improvement of the claims

of bike patent 1. Row 5414 indicates that the claims of bike patent 1 are dominant over the claims of bike patent 2. Rows 5410-5414 are useful for generating an assignee patent family chronology claim tree, described above.

Operational Description

5 FIG. 55 illustrates a flowchart 5502 that represents the operation of the invention when generating data object family tables. Depending on the particular relationships of interest, the sources of relationship data, the extent of the families of interest, and other application dependent factors, the data object family tables may be manually generated, automatically generated, or
10 combinations thereof.

 In step 5506, relationship information is obtained. Preferably, relationship information is obtained in a pair wise fashion. In other words, pairs of data objects that are somehow related are first identified. Then, the relationships between the data object pairs are determined. Alternatively,
15 relationship information is obtained and then converted into a pair wise format.

 There are a variety of sources of relationship information. Some of these sources applicable with the present invention are illustrated in FIG. 56. It is noted that the invention is not limited by the sources depicted in FIG. 56. Relationship information may be available and obtained from other sources,
20 including but not limited to data sources identified and/or provided by users.

 One source of relationship information is data having referential integrity, as indicated by 5604. Referential integrity refers to information that directly indicates a relationship between data objects. For example, the information in the patent ref table 4402 of FIG. 44 has referential integrity because the
25 information contained therein directly establishes citation relationships between patents. Similarly, the non-patent ref table 4404 in FIG. 45 has referential

integrity because the information contained therein directly establishes citation relationships between patents and non-patent data objects.

Referring to FIG. 56, relationship information can also be obtained from relational database tables 5606. By definition, relational databases relate data objects to one another. Accordingly, the relationship represented in relational databases can be used to generate relationship tables and data object family tables.

Relationship information can also be obtained by searching database tables, as indicated by 5608 in FIG. 56. Such operations may include, for example, searching a "corporate entity" column of a database table for all occurrences of the name "Aurigin," to thereby identify all data objects that relate to the company Aurigin. This might result in identifying all patents that are assigned to Aurigin. This information is useful for generating patent family chronologies, for example.

Relationship information can also be obtained by performing text searches among data objects. Such operations may involve, for example, searching the text of documents for the word "microprocessor" to thereby identify data objects that potentially relate to the microprocessor field. This information is useful for generating technology patent families, for example. Searches could include more complex Boolean, proximity, natural language, and any types of linguistic analysis.

Additional relationship information may be obtained by performing manual acquisition and analysis of data, as indicated by 5612 in FIG. 56. Such operations may involve, for example, manually reviewing the claims of a collection of patents to determine those that are technically related, dominant and subservient relationships, base relationships, improvements, etc.

Further relationship information can be obtained by combinations of the techniques illustrated in FIG. 56.

Upon completing step 5506, data object pairs having some relationship of interest have been identified, and the relationships between such data object pairs have been determined. In step 5508, data object family tables are populated using the relationship information obtained in step 5506. For example, suppose
5 that in the course of performing step 5506 it was determined that a particular Japanese application, called Japan 1, was filed from a particular PCT application, called PCT 1. In this case, a row in a data object family table corresponding to this pair wise relationship is created in step 5506. Such a row is shown, for example, as 5216 in FIG. 52.

10 Upon completing step 5508, one or more data object family tables have been created.

FIG. 57A illustrates a flowchart 5702 for analyzing relationship information according to embodiments of the invention. For illustrative purposes, and without limitation, flowchart 5702 is described with reference to
15 the example patent family chronology 5102 shown in FIG. 51, and the corresponding data object family table 5202 shown in FIG. 52.

In step 5706, the user selects one or more data objects. The selected data objects represent the base data objects upon which the analysis is performed. In the example of FIG. 51, the user selects application 1, which is the base patent
20 application for the patent family chronology 5102.

In step 5708, the user selects one or more relationship types upon which the analysis is formed. In an embodiment, the user manually selects the relationship types of interest. In other embodiments, the user selects one or more grouping that contain the relationship types of interest. In the example of FIG.
25 51, the user selects grouping G1 (see FIGS. 50A and 50B), which contains all of the relationship types of interest for the patent family chronology (it is noted that grouping G1 can be modified as desired by the user to perform a custom patent family chronology analysis).

5 In step 5710, the user indicates whether complete or partial closure is required. If the user selects complete closure, then all relationships stemming from the data objects selected in step 5706 are identified. If the user selects partial closure, then only a portion of the relationships stemming from the selected data objects are identified (for example, perhaps only relationships moving forward or moving backward from the selected data objects are identified). Also in step 5710, if the user selects partial closure, then the user may be required to provide information regarding the scope of the desired analysis.

10 In step 5712, closure based on the selected data objects, the selected relationship types, and the scope of closure is computed. In other words, all relationships from the data object family tables that satisfy the information and commands provided in steps 5706, 5708, and 5710 are identified.

15 Preferably, step 5712 is performed as shown in the flowchart of FIG. 57B.

20 In step 5718, rows in the data object family tables that match the selected data objects and the selected relationship types are identified. A row matches a selected data object if either the "from" field or the "to" field of the row contains the selected data object. A row matches a selected relationship type if the relationship type field of the row contains one of the selected relationship types.

25 Step 5720 is optionally performed depending on the level of closure specified in step 5710. In step 5720, additional rows in the data object family tables that match data objects specified in rows identified in step 5718, and that also match one or more of the selected relationship types are identified. For example, in the data object table 5202 of FIG. 52, row 5206 is identified in step 5718 because application 1 (the data object selected in step 5706) is contained in the "from" field. Note that continuation 1 is contained in the "to" field. In step 5720, rows in the data object family tables that match continuation 1, and that also match one or more of the selected relationship types are identified. Thus,

row 5212 is identified because continuation 1 from row 5206 is contained in the "from" field, and also the trademark relationship (which is part of selected group G1) is contained in the relationship type field.

Steps 5718 and 5720 are repeatedly performed until the level of closure specified in step 5710 is satisfied, as indicated by control flow line 5721.

Upon the completion of steps 5718 and 5720, data object pairs having relationships of interest have been identified. Also, the relationships between such data object pairs are identified. In step 5722, a tree of the data object pairs is generated. Preferably, the tree is generated by selecting a data object pair, adding the selected data object pair and its associate relationship to the tree, and then selecting a new data object pair. The process continues until each data object pair has been added to the tree.

For example, assume that steps 5718 and 5720 identified rows 5204-5218 in FIG. 52. In step 5722, the tree or graph is generated from rows 5204-5218. For example, processing of row 5204 results in creating nodes 5106 and 5108 and link 5104 in the tree (see FIG. 51). Processing of row 5206 results in adding node 5110 and link 5112 to the tree (FIG. 51). The other identified rows 5208-5218 are processed in a similar manner in step 5722.

Referring again to FIG. 57A, in step 5714, the data object family is displayed. According to the invention, the tree generated in step 5722 (FIG. 57B) may be displayed using any visualization technology, including hyperbolic trees. Techniques for generating hyperbolic trees are discussed in U.S. patent application "Using Hyperbolic Trees to Visualize Data Generated by Patent-Centric and Group-Oriented Data Processing," Ser. No. 08/921,369, incorporated herein by reference in its entirety.

Plug-In Architecture

According to the invention, the enterprise server 7604 (FIG. 76) can interface with third party tools to enhance and extend its functionality. As used herein, the phrase "third party tools" refers to any hardware, software, or combination thereof that is external to the enterprise server 7604 or any of its component or associated modules, such as the web server 7610, the databases 7612 and 7614, the network client 7608, and the web client 7606. The third party tools are selected and integrated with the enterprise server 7604 on the basis of their ability to perform desired functionality not available internally (i.e., not performed by the enterprise server 7604 or any of its components or associated modules). Such functionality can pertain to data acquisition or storage, visualization, display, processing, and/or any other data processing.

FIG. 59 illustrates a flowchart 5902 representing illustrative operation of the enterprise server 7604 when interacting with a third party component.

In step 5906, a user begins analyzing a collection of data objects. The data objects may be, for example, from one or more groups, and/or one or more data object types. Alternatively, the data objects may have resulted at least in part from processing performed by the enterprise server 7604, such as from conducting database searches.

In step 5908, the user analyzes the data objects using functions provided by the enterprise server 7604. Such functions are described above. Additional functions are described in U.S. patent applications "System, Method, and Computer Program Product for Patent-Centric and Group-Oriented Data Processing," Ser. No. 08/867,392, "Using Hyperbolic Trees to Visualize Data Generated by Patent-Centric and Group-Oriented Data Processing," Ser. No. 08/921,369, "System, Method and Computer Program Product for Managing and Analyzing Intellectual Property (IP) Related Transactions," Ser. No. 09/138,368, incorporated herein by reference in their entireties.

In step 5910, the user wishes to run one or more reports based on the analysis performed in step 5908. Some of the reports may be generated using

functions provided by the enterprise server 7604. Such reporting functions of the enterprise server 7604 are described herein, and are further described in the patent applications referenced above. Other reports may be generated using third party components that interface with the enterprise server 7604.

5 In step 5912, the user wishes to further analyze the data objects using functions provided by third party components that interface with the enterprise server 7604. Information pertaining to the data objects are sent to the third party components, along with commands, options, and other information needed to allow the third party components to perform the analysis requested by the user.
10 The third party components perform the requested functions using the information received from the enterprise server 7604.

In step 5914, the user instructs the third party components to send the results of their respective processing back to the enterprise server 7604. Such results may be stored and utilized as a new group or data object type in the enterprise server, for example. Such results could also be processed alone or
15 with other data, and/or could be stored as data in fields of data records and/or reports.

FIG. 64 is a flowchart 6402 further depicting the operation of the enterprise server 7604 when interacting with a third party component according to an example embodiment. FIG. 63 is an example data flow diagram used to
20 explain the operation of the flowchart 6402 in FIG. 64.

In step 6406, the user wishes to analyze a group 6304 containing one thousand documents or other data objects. The documents include both patents and non-patent documents. The documents relate to computer programming
25 languages.

The user in step 6406 selects a third party component to organize and visualize the content of the documents. The display generated by the selected third party component is shown as 6306 in FIG. 63. In display 6306, documents that are similar in content are represented as peaks in a landscape. The relative

size of the peaks correspond to the number of documents that are directed to the subject matter associated with the peaks. Preferably, operation of the third party component is dynamic, such that the context of the landscape is dynamic. For example, the display 6306 shown in the example of FIG. 63 is based on the subject matter or topic of the documents in the group 6304. Preferably, the user can access a menu 6308 which allows the user to change the context of the display from topical to another attribute, such as author. If the user changes the context to author, then the peaks in the display 6306 would correspond to author, and the relative size of the peaks would indicate the number of documents that were by respective authors.

In step 6408, the user selects a subset of the 1,000 documents via use of the third party component. In the example of FIG. 63, the user has selected from the display 6306 the peaks corresponding to PASCAL and COBAL. This selection is indicated by 6307.

In step 6410, the user instructs the third party component to send the results of the analysis back to the enterprise server 7604. In the example of FIG. 63, the third party component sends back an indication of the selection 6307. For example, the third party component may send back a list of the documents corresponding to the selection 6307 (that is, a list of the books directed to PASCAL and COBAL).

In step 6412, the enterprise server 7604 creates a new group from the information received from the third party component. In this case, the new group contains the documents corresponding to the selection 6307 that pertain to PASCAL and COBAL.

In step 6414, the enterprise server generates and saves a search query corresponding to the analysis performed by the user using the third party component. In example of FIG. 63, the search query generated and saved by the enterprise server 7604 comprises a search for documents pertaining to PASCAL

and COBAL. The search query is generated and saved so that it can be referred to and used in the future by the user.

5 The new group 6310 can then be used in the enterprise server 7604 just like any other group. Specifically, processing can be performed on the documents in the new group 6310 as described above, and as further described in U.S. patent applications "System, Method, and Computer Program Product for Patent-Centric and Group-Oriented Data Processing," Ser. No. 08/867,392, "Using Hyperbolic Trees to Visualize Data Generated by Patent-Centric and Group-Oriented Data Processing," Ser. No. 08/921,369, "System, Method and 10 Computer Program Product for Managing and Analyzing Intellectual Property (IP) Related Transactions," Ser. No. 09/138,368, herein incorporated by reference in their entirety.

15 The examples discussed above illustrate the round trip functionality of the present invention. Specifically, the invention allows information to be transferred from the enterprise server 7604 to third party components. Also, the invention enables the transfer of information from third party components to the enterprise server 7604. In certain instances, depending on the functionality desired and the third party component being used, only one way interaction is involved. For example, often when working with a report generator, data flows 20 only from the enterprise server 7604 to the report generator. Such data is sent to the report generator to enable the report generator to create a desired report. With some report generators, data does not flow back from the report generator to the enterprise server 7604 (although, with some report generators, data may flow back to the enterprise server 7604. For example, the report generator may 25 send an electronic copy of the report back to the enterprise server 7604).

Plug-In Architecture

FIG. 58 is a block diagram 5802 of the enterprise server 7604 when interacting with third party components/tools 5808. As indicated in FIG. 58, the enterprise server 7604 interacts with third party components/tools 5808 via an interface 5806.

5 FIG. 60 illustrates a more detailed block diagram of the enterprise server 7604 when interacting with third party components/tools 5808.

 The enterprise server 7604 includes an enterprise server API (application programming interface) 6004. Generally, the enterprise server API 6004 includes a collection of commands that the enterprise server 7604 understands. These commands instruct the enterprise server 7604 to perform specific functions. An entity external to the enterprise server 7604 interacts with the enterprise server 7604 by sending commands that conform to the enterprise server API 6004 to the enterprise server 7604. The enterprise server API 6004 is further described below. The enterprise server API 6004 is additionally described in U.S. patent application U.S. Patent Application "System, Method, and Computer Program Product for Patent-Centric and Group-Oriented Data Processing," Ser. No. 08/867,392, incorporated herein by reference in its entirety.

 The interface 5806 is preferably represented by a plug-in manager 6006. The plug-in manager 6006 includes a plug-in manager API 6008. The plug-in manager API includes commands which the plug-in manager 6006 understands. These commands instruct the plug-in manager 6006 to perform specific functions. Entities external to the plug-in manager 6006 interact with the plug-in manager 6006 by sending commands that conform to the plug-in manager API 6008 to the plug-in manager 6006.

25 The third party components/tools 5808 are represented by plug-ins 6012 and third party components/tools 6024. The third party components/tools 6024 represent any hardware, software, or combination thereof that performs desired functionality. The third party components/tools 6024 each includes an API.

Devices external to the third party components/tools 6024 interact with the third party components/tools 6024 via their respective APIs.

Preferably, a plug-in 6012 exists for each third party component/tool 6024. Alternatively, one or more third party components/tools 6024 may share a common plug-in 6012.

Generally, the plug-ins 6012 represent translation modules. Messages sent from the plug-in manager 6006 to a third party component/tool 6024 is translated from a format native to the plug-in manager 6006 to a format recognized by the API of the third party component/tool 6024. Similarly, a message that is send from a third party component/tool 6024 to the plug-in manager 6006 is translated by the plug-in 6012 from the native language of the third party component/tool 6024 to commands and/or messages that conform to the plug-in manager API 6008.

As indicated in FIG. 60, the enterprise server 7604, the plug-in manager 6006, the plug-ins 6012, and the third party component/tools 6024 interact with a database 6010, which may include, for example, the IPAM databases 7614.

The process of translating between the plug-in manager API 6008 and the API of any given third party component/tool 6024 will be apparent to persons skilled in the art based on the discussion of the plug-in manager API 6008 contained herein, and knowledge of the well known API of the third party component/tool 6024. Additionally, interaction with a third party component/tool 6024 via its API will be apparent to persons skilled in the relevant arts.

The plug-in architecture described herein is applicable to both the server and the client. In embodiments of the invention, the client plug-in architecture works in a manner substantially similar to the server plug-in architecture, a difference being in the type and kind of APIs supported/required by the client and the manner in which the client detects that plug-ins are present. Implementation

of the server and client plug-in architectures will be apparent to persons skilled in the relevant art(s) based on the discussion contained herein.

Plug-In Operation

FIGS. 62A and 62B collectively illustrate a flowchart 6202 representing the operation of the plug-in manager 6006 when interacting with a third party component 6024 according to an embodiment of the invention. FIG. 61 illustrates an example event trace diagram corresponding to the flowchart of FIG. 62A and 62B.

In step 6206, the enterprise server 7604 creates an instance of the plug-in manager 6006. This is indicated as 6120 in FIG. 61.

In step 6208, the plug-in manager determines which plug-ins 6012, if any, are available. This is indicated as 6122 in FIG. 61. As noted above, each plug-in preferably corresponds to a third party component 6024, and understands the API, the information requirements, the options, etc., of the corresponding third party component 6024 (i.e., how to interact with the third party component).

In step 6210, the enterprise server receives from the plug-in manager 6006 a list of the available plug-ins 6012. This is indicated as 6122 in FIG. 61. This list indicates which third party components 6024 are available for use by the user.

In step 6212, the user pops up a plug-in menu 6104. The menu 6104 lists all plug-ins 6012 that are available based on the list obtained in step 6210. Further in step 6212, the user selects a plug-in from the plug-in menu 6104.

In step 6214, the enterprise server 7604 instructs the plug-in manager 6006 to run the selected plug-in. This is indicated by 6124 in FIG. 61.

In step 6218, the plug-in manager 6006 instructs the selected plug-in 6012 to run. This is indicated by 6128 in FIG. 61.

In step 6220, the plug-in 6012, possibly with the assistance of the plug-in manager 6006, displays an options dialog or a series of dialogs to the user. The dialogs specify the functions that can be performed by the corresponding third party component 6024. Preferably, at least all functions that are accessible through the third party component's API are listed in the dialogs. In other words, interaction between the enterprise server 7604 and the third party component 6024 includes at least the functions and capabilities represented in the API of the third party component 6024. The user selects desired options via the displayed options dialog. This interaction is represented by 6130 in FIG. 61.

In step 6222, the plug-in 6012 makes requests for data from the plug-in manager 6006. This is represented by 6138 in FIG. 61. In response to such requests, the plug-in manager 6006 may make calls to the enterprise server 7604 (represented by 6134 in FIG. 61), or may directly access the databases 6010 (this is represented by 6136 in FIG. 61).

In step 6224, the plug-in 6012 creates an instance of the associated third party component 6024. This instance is a software representation of the third party component. For example, if the third party component is a report generator, then in step 6224 the plug-in invokes the report generator. This is represented by 6140 in FIG. 61.

In step 6226, the plug-in transfers data to the third party component 6024. The plug-in performs any necessary data transformations. Additionally, the plug-in 6012 sends commands and options (from step 6220) to the third party component 6024, or alternatively controls the third party component 6024 based on the commands and options from step 6220. Preferably, the plug-in 6012 interacts with the third party component 6024 in this manner using the API of the third party component 6024. This operation of step 6226 is represented by 6142 in FIG. 61.

Also in step 6226, the user performs any real time interaction with the third party component 6024 via the plug-in manager 6006 and the plug-in 6012.

An example of such interaction was described above with reference to FIGS. 63 and 64.

In step 6228, the user instructs the third party component 6024 to export the results of any processing performed by the third party component 6024. This is represented by 6144 in FIG. 61.

In step 6230, in response to the export command, the third party component 6024 transmits data to the enterprise server 7604 via the plug-in 6012 and the plug-in manager 6006. The plug-in 6012 and/or the plug-in manager 6006 perform any necessary data transformations. The enterprise server 7604 receives and processes the information as described above.

Interaction with third party components shall now be considered in greater detail.

Embodiments of the invention support a number of methods for integration with third party components. Two such integration methods are called "loose integration" and "tight integration." These are described below.

Loose Integration

With loose integration, interaction is via an intermediary, such as files. To export data from application 1 (such as IPAM) to application 2 (such as a third party tool), application 1 or an entity working on behalf of application 1 stores the data in a file (preferably in a format understandable by application 2; otherwise, data translations may be performed). The file is then read by application 2, or an entity working on behalf of application 2.

To import data to application 1, application 2 or an entity working on behalf of application 2 saves data in a file. The file is then read by application 1, or some entity working on behalf of application 1 (some data translation may be necessary). In embodiments of the invention, these write and read operations are

manually initiated. That is, the user has to manually initiate the exportation and importation of data.

Generally, it is possible to utilize off-the-shelf versions of a third party component when loose integration is used.

5

Tight Integration

With tight integration, IPAM and a third party component interact with each other via the plug-in architecture described herein. Data is sent from IPAM to a third party component via the plug-in mechanism, and visa versa.

10

In an embodiment, IPAM (via the plug-in mechanism) sends a list of pointers to the third party component. The pointers point to data (i.e., data objects). In this manner, IPAM sends data to third party components. This same process is also used to send data from third party components to IPAM (via the plug-in mechanism). Alternatively, IPAM could send data to a third party component by storing the data in a file, and then sending a message to the third party component (via the plug-in mechanism) with the name of the file. This process could also be used to send data from third party components to IPAM. Alternatively, IPAM could send data to a third party component by sending messages (containing the data) via the plug-in mechanism to the third party components. This process could also be used to send data from third party components to IPAM.

15

20

25

Tight integration is a more automated process. The user need not explicitly store data in files, or manually initiate read and/or write operations. Instead, the user need only issue import or export commands. An example of the user issuing an export command is shown, for example, in FIG. 90. In the example of FIG. 90, the user is exporting the data objects in a group "adewolfe" to Microsoft Excel. The menu 9002 shown in FIG. 90 also illustrates other commands 9004 to import and export.

5 In order to perform tight integration, it is necessary that the API of the third party component include sufficient functionality to support the interactions described above. Alternatively, tight integration can be achieved using a version of a third party component that has been modified to support the interactions described above.

Query Creation and Retention

10 According to the invention, upon receipt of information from a third party component, the enterprise server 7604 creates a search query that corresponds to the analysis/processing performed by the user using the third party component. In an embodiment, the enterprise server 7604 generates the query by comparing the information sent to the third party component with the information received from the third party component.

15 Consider the example of FIG. 63, where 1000 documents pertaining to computer programming languages were sent to the third party component, and a subset of these documents pertaining to Pascal and Cobol were returned to the enterprise server 7604. According to an embodiment, the enterprise server 7604 compares what was sent with what was received to generate a search query limited to Pascal and Cobol.

20 In other embodiments, the third party component returns status information that the enterprise server 7604 uses to generate the search query. In other embodiments, the third party component returns the search query.

The query is saved for later reference and use by the user.

Plug-in Manager Application Program Interfaces (APIs)

25 The Plug-in Manager is preferably an ActiveX component that supports the addition of "plug-ins" (also preferably ActiveX components) that support

interaction with third-party components. As noted above, plug-ins are used to export data to and import data from third-party visualization components (such as but not limited to Microsoft Excel, Cartia ThemeScape, and SpotFire).

5 The Plug-in Manager defines several APIs, described below, for use by various parts of the Plug-in facility. The Plug-in Manager also provides a local database, which provides "linked tables" (proxies) to tables in the IPAM SQL Server database.

10 When the IPAM is started, the IPAM queries the Plug-in manager for available Plug-ins, preferably creating a pull-right menu listing their names. (If no Plug-ins are available, the menu is not created, and access to Plug-ins is disabled in the IPAM.) In one embodiment, a user selects an IPAM group, then uses the Plug-in menu to cause the Plug-in Manager to invoke a specific Plug-in. Preferred protocols by which this happens are described below.

15 Additional APIs applicable with the invention are described in "System, Method, and Computer Program Product for Patent-Centric and Group-Oriented Data Processing," Ser. No. 07/867,392, filed June 2, 1997, which is incorporated herein by reference in its entirety. Implementation of the APIs and commands described below will be apparent to persons skilled in the relevant art(s) based on the teachings contained herein.

20 It should be understood that the following APIs and commands are provided for illustrative purposes only, and are not limiting. Any other APIs or commands that achieve the functions and requirements described herein can also be used, as will be apparent to persons skilled in the relevant art(s) based on the herein teachings.

25 ***API: IPAM to Plug-in Manager***

The Plug-in Manager exposes an API to the IPAM. This API allows the IPAM to obtain the number and names of available Plug-ins, and to invoke those Plug-ins, but preferably shields the IPAM from Plug-in-specific details.

In an example embodiment, plug-ins advertise their availability by way of a Windows Registry subkey under the key:

HKEY_LOCAL_MACHINE\Software\Aurigin Systems, Inc.\Aurigin
Client\Plug-ins\

The name of the subkey is the OLE class name of the Plug-in (e.g., Aurigin.ExcelPlug-in).

If the default value of the subkey is "0", then the Plug-in is considered to be unavailable. Any other value indicates that the Plug-in is available. In other embodiments, the Plug-in Manager may automatically mark a Plug-in as being unavailable under certain circumstances. (For example, if a Plug-in fails some set number of times in succession).

The "Name" value of the subkey holds the name of the Plug-in, as it is to appear in IPAM menus.

The "Description" value of the subkey holds a string that provides additional information about the Plug-in. For example, to be used as a tooltip.

Various example commands/functions are described below.

Plug-inCount() As Integer

Returns the number of available Plug-ins.

The Plug-inManager queries the Registry to identify available Plug-ins, as described above.

Plug-inName(index As Integer)

Returns the name of the index'th Plug-in, where $0 \leq \text{index} < \text{Plug-inCount}()$.

Plug-inDescription(index As Integer) As String

5 Returns the description of the index'th Plug-in, where $0 \leq \text{index} < \text{Plug-inCount}()$.

SetGroup(name, id As String)

Sets the name and database id of the group that will be the source of data for a Plug-in.

As a side-effect, the list of subgroups and the list of documents is cleared.

10 ***AddDocument(name, id As String)***

Adds the name and id of a document to the list of documents in the group. After the IPAM invokes SetGroup(), the IPAM is required to invoke AddDocument() once for each document in the group.

SetAttribute(name, value As String)

15 Sets or resets a dictionary entry for "name" to be "value". The dictionary can be queried by a Plug-in using protocol described further below. The IPAM is required to set the following attributes

UserId the ID of the logged-in user

IPAMServer the name and port number of the IPAM server (e.g.,
20 "Romulus:8080")

SQLServer the name of the server on which SQL Server is running

RunPlug-in(Integer index)

Run the index'th Plug-in, where $0 \leq \text{index} < \text{Plug-inCount}()$.

5 The Plug-in manager creates an instance of the ActiveX component for the Plug-in, and invokes it using the "Plug-in Manager to Plug-in" API, described below.

If the Plug-in "fails" (where conditions constituting failure are defined by the specific Plug-in), the Plug-in will raise an OLE exception. If no exception is raised, the Plug-in is assumed to have succeeded.

10 ***API: Plug-in Manager to Plug-in***

Each Plug-in preferably implements this API, through which the Plug-in Manager invokes (runs) the Plug-in.

Various example commands/functions are described below.

Run(Object manager)

15 Run the Plug-in. The "manager" object is a reference to the instance of the Plug-in Manager ActiveX component. The Plug-in uses the "Plug-in to Plug-in Manager" API, described below, to communicate with the Plug-in manager.

When run, a Plug-in preferably will do one or more of:

- 20 *
- * present a dialog to the user listing Plug-in-specific options
- * query the Plug-in Manager for the selected group and its documents

- * direct the Plug-in Manager to perform citation traces for documents
- * query a local database for additional information about documents
- * invoke a third-party API to communicate with a third-party application or component (e.g., Microsoft Excel, Cartia ThemePublisher, SpotFire, Inxight Hyperbolic Tree)
- * insert new data into a local database
- * invoke the IPAM "Plug-in to IPAM" API to create new groups of documents
- * present one or more status of informational dialogs to the user

If the Plug-in encounters an error, it communicates details back to the Plug-in Manager by raising an OLE exception, passing a description of the error in the exception.

API: Plug-in to Plug-in Manager

- 15 The Plug-in Manager implements this API for use by Plug-ins.
Various example commands/functions are described below.

GetAttribute(name As String) As String

- 20 Get the value of an attribute, as set by SetAttribute(). The Plug-in uses this function as necessary to obtain the id of the logged-in user, the name of the server name and port number of the IPAM server, etc.

Raises an exception if the name is unknown.

UnrollGroups(depth As Integer)

IPAM Groups form a hierarchy. UnrollGroup populates the bds_groups table in the local database with a trace of the group hierarchy from the source group for the given depth. (A depth of 0 means the source group only, 1 means to include the source group and one level of subgroups, and so on.)

5 The Plug-in then gains access through the GetDB() function, described below, and can issue SQL queries that join against the bds_groups table.

GroupCount() As Integer

Returns the number of source groups. In an embodiment, this always returns 1.

10 ***GetGroups(name() As String, String id() As String)***

Gets the names and ids of groups.

The Plug-in is responsible for sizing the name and id arrays to accommodate GroupCount() groups.

DocumentCount() As Integer

15 Returns the number of documents in the source group(s).

GetDocuments(name() As String, id() As String)

Gets the (long display) names and ids of documents in the source group(s).

The Plug-in is responsible for sizing the name and id arrays to accommodate DocumentCount() documents.

20 ***TraceCitations(forward As Boolean, depth As Integer)***

Directs the Plug-in manager to trace citations for the source documents, writing the results into the 'bds_citations' table in the local database. Citations are traced either forward or backward, and to a given depth.

5 The Plug-in then gains access to the local database via GetDB(), as described below, and can issue SQL queries that join against the bds_citations table.

GetDB() As Database

10 Return an ActiveX object that provides access to the local database. "Database" is an OLE Object type provided by Microsoft. It provides an API for interacting with JET databases. The database object has an API that supports SQL queries.

Through the local database, the Plug-in gains access to tables in the IPAM database, and optionally to third-party or customer-specific tables, allowing information to be merged for presentation.

API: Plug-in to IPAM

15 The IPAM exposes an API for creating new groups.

Various example commands/functions are described below.

CreateGroupFromGUIDs(name, description As String, guid() As String)

20 Create a new group with the given name and description, populating it with documents.

Preferably, use of this API requires that:

1. a user be logged in to the IPAM
2. a group is selected in the group pane
3. the user has permission to create a new subgroup within the selected group

5 If the guid() array contains ids that are not recognized, those ids are written to a log file.

Example Third Party Components

10 The enterprise server 7604 can interact with any third party component having functionality and capabilities of interest. As indicated above, the enterprise server 7604 interacts with a given third party component 6024 via the API of the third party component 6024. Alternatively, the enterprise server 7604 interacts with a third party component 6024 via a custom interface to the third party component 6024.

15 For illustrative purposes, and without limitation, example third party components 6024 are described below. It should be understood that the present invention is not limited to working with these third party components. The following discussion is provided solely to illustrate example operation of the enterprise server 7604 with third party components 6024, and is not limiting.

Graph-Based Visualization Tools

20 FIG. 67 illustrates an example display where data is represented in a graph. The example of FIG. 67 depicts a two-dimensional graph, but graphs of dimensions greater than two are within the scope and spirit of the invention.

25 In the example display 6702 of FIG. 67, each icon in the graph 6708 represents one or more patents. In an embodiment, information pertaining to a patent (such a bibliographic information, text information, image information,

etc.) can be retrieved and displayed by selecting the icon associated with the patent.

The X-axis of the graph represents the issue date, and the Y-axis of the graph represents the filing date. The patent icons are positioned on the graph 6708 depending on their particular filing dates and issue dates. The display can be dynamically adjusted by selecting other criteria for the X-axis via a drop down menu 6710. Similarly, the display can be dynamically adjusted by selecting other criteria for the Y-axis via a drop down menu 6712.

The data represented in the graph 6708 can be filtered by selecting from attributes 6706. In example of FIG. 67, for example, the patents represented in the graph 6708 are limited to art units 111 through 358, and remaining pendency of 42 weeks to 961 weeks.

A variety of graph based visualization tools operable with the present invention are publicly available. For example, and without limitation, SPOTFIRE PRO is a commercial product available from SPOTFIRE, Boston, Massachusetts. SPOTFIRE PRO enables the display of data in graphical form as shown in FIG. 67.

Landscape-Based Visualization Tools

FIG. 65 illustrates an example display 6502 wherein data is represented in a landscape or map. Peaks in the landscape correspond to data objects that have been aggregated according to user selected criteria. In the example of FIG. 65, the peaks correspond to claims from one or more patents that are directed to similar subject matter. For example, reference number 6508 identifies a peak that represents an aggregation of claims from one or more patents that are directed or related to "policy". The relative heights/size of the peaks indicate the number of claims corresponding to the respective subject matters of the peaks.

The display can be dynamically adjusted by selecting attributes from a menu 6510. For example, if the "assignee" attribute is selected, then the landscape would be adjusted so that the peaks would represent the number of patents assigned to corporate entities.

5 Information in the display can be limited by using filters from a filter menu 6512. For example, if an "assignee" attribute is selected, and "Aurigin" is entered in a pop-up dialog, then the landscape would be adjusted to only display patents assigned to "Aurigin."

10 According to the example shown in FIG. 65, when the cursor 6504 is positioned proximate to or above a peak, the text of a claim that corresponds to the peak is partially or fully displayed. This is illustrated by window 6506.

FIG. 66 illustrates a similar display 6604 that represents patent related data using a landscape approach.

15 FIG. 63, described above, is directed to another display 6306 wherein information is visualized using a landscape approach.

As described above, and further illustrated in FIG. 88, information from IPAM 8806 can be exported to the landscape tool 8808. This is represented by data flow line 8802. The exported information is represented as a map 8810. The user can manipulate the map to select groups of data objects, which can then be
20 imported to IPAM 8806. This is represented by data flow lines 8804. The imported data can be used to form new groups, as shown in FIG. 88.

25 An example product that utilizes a landscape approach to visualize data is THEMESCAPE from CARTIA, INC., Bellevue, Washington. FIG. 68 illustrates an event diagram 6802 representing the interaction between the enterprise server 7604, the enterprise client 7606 or 7612, and the third party THEMESCAPE product 6807 according to an embodiment of the invention. The invention is not limited to this example implementation.

Interaction with Cartia ThemeScape will be further illustrated by reference to an example scenario shown in FIGS. 91-98. This also further illustrates the round-trip capabilities of the invention.

First, an IPAM group is selected, and this is sent to Cartia ThemeScape. See FIG. 91.

Second, available options are chosen in the Cartia Plugin dialog (see FIG. 92). The dialog allows you to change the name of the map (the name of the IPAM Group is the default map name). The dialog also allows you to specify what portions of the patent documents in the group to use to build the map. Dialogs vary among third party tools.

Third, a notification is provided indicating that Cartia has begin processing the map. See FIG. 93.

Fourth, progress is indicated in the ThemePublisher's Map Manager dialog. See FIG. 94. When it has built the map, the status line will say "Processing Complete", or will indicate why the map could not be built.

Fifth, the map is displayed. See FIG. 95. The user uses ThemePublisher's tools to select a portion of the map.

Sixth, with a portion of the map selected, the ThemePublisher Document Viewer appears. See FIG. 96. The user clicks the Export button to export the data objects corresponding to the selected map portion to IPAM.

Seventh, an Export dialog is presented. See FIG. 96. In the example of FIG. 96, the "Aurigin IPAM Group" option is selected, and then the user clicks OK. In some embodiments, Cartia ThemePublisher displays a dialog that asks, "What should the Aurigin IPAM group be named?" with a field for the user to enter a name. It will contain a default value such as the map name.

Eighth, a new group is created as the child of whatever group is selected.

This example has been provided for purposes of illustration, not limitation. The example is applicable to other third party tools. Accordingly, given the herein teachings, persons skilled in the relevant art(s) could integrate

other third party tools with IPAM. The invention is directed to such other combinations.

Data Object Searching Tools

5 The invention is operable with third party tools for identifying collections of data objects. Such data objects can be imported into IPAM for processing. For example, one or more new groups may be formed from the imported data objects.

10 For example, FIG. 89 illustrates a chemical search engine 8904. (The chemical search engine 8904 is herein described for illustrative purposes only. The invention is not limited to this embodiment.) The user defines a chemical structure 8905 in the chemical search engine 8904. The chemical search engine 8904 then searches for chemicals that are related to the chemical structure 8905. A variety of chemical search engines are publicly available and operable with the
15 invention, such as CAS SciFinder.

 The results of the search are stored via a save dialog 8906. In the example of FIG. 89, the search results are saved in file "test.csv." The search results comprise one or more chemicals that were found during the search. Generally, each chemical represents a data object.

20 In IPAM 8902, the user imports the file "test.csv," as indicated by arrow 8908. The contents of "test.csv" (i.e., the chemicals located during the search) are stored in a group 8910.

 An example use scenario is as follows.

25 A user working in IPAM 8902 is studying a group comprising a multitude of chemical compounds (each compound may be represented by a corporate document, for example). The user wishes to identify chemical compounds in this group that have a particular sub-structure.

The user instructs IPAM 8902 to invoke the chemical search engine 8904, and export the group to the chemical search engine 8904. Accordingly, the plug-in manager interacts with the plug-in for the chemical search engine 8904 to (1) invoke the chemical search engine 8904, and (2) export the group to the chemical search engine 8904. The plug-in interacts with the chemical search engine 8904 using the API of the chemical search engine 8904. The plug-in performs any necessary data/format translations to export the group to the chemical search engine 8904.

The user interacts with the chemical search engine 8904 to conduct a search for compounds related to a particular structure 8905, as described above. This search is limited to the compounds in the group that was exported to the chemical search engine 8904.

After completion of the search, the plug-in associated with the chemical search engine 8904 accesses the search results (possibly from file "test.csv") and imports the search results to IPAM 8902 via the plug-in manager. In doing so, the plug-in performs any needed data/format translations consistent with the API of the plug-in manager.

IPAM 8902, upon receiving the search results, may create a new sub-group of the original group, and store the search results in the new sub-group. Other processing by IPAM 8902 is possible, as will be apparent to persons skilled in the relevant art(s) based on the teachings contained herein.

Other

Additional third party components applicable for use with the enterprise server 7604 include, but are not limited to, CRYSTAL REPORTS from SEAGATE, EXCEL from MICROSOFT, TABLE LENS from INXIGHT, and PERSPECTA from PERSPECTA, INC.

Hybrid Models

The invention supports a variety of configurations of its components. Additionally, the invention supports a variety of functional configurations and distributions.

5 For example, FIG. 69 illustrates a configuration where all primary components and functions are positioned at a single location 6904.

FIG. 70 illustrates a configuration where components and functionality are distributed among a first location 7002 and a second location 7004. The enterprise server 7604, the client 7606, 7608, and the IPAM databases 7614, and functions performed by these components, are positioned at the first location 10 7002. The search engines and indexes 6902, and the functions performed by the search engines and indexes 6902, are positioned at a second location 7004. To perform a search specified by the user, the enterprise server 7604 accesses the remote search engines and indexes 6902 at the second location 7004. It is noted that such search engines and indexes 6902 may include third party components, 15 such as LEXIS-NEXIS, Westlaw, Dialog, Derwent, etc., as described above.

FIG. 71 illustrates a configuration where components and functionality are distributed among a first location 7102, a second location 7104, and a third location 7106.

20 FIG. 72 illustrates a configuration where components and functionality are distributed among a first location 7202, a second location 7204, a third location 7206, and a fourth location 7208.

As indicated by the example configurations illustrated in FIGS. 69-72, the present invention includes a variety of configurations and distributions of its components and functionality. The examples of FIG. 69-72 are presented for 25 purposes of example only and are not limiting. Other configurations of components and functionality will be apparent to persons skilled in the relevant arts based on the discussion contained herein.

For example, a given component and/or functionality can be distributed among multiple locations. This is illustrated in FIG. 73, where the enterprise server 7604 is distributed among a first location 7302 and a second location 7304. Additionally, databases 7206 are distributed among location 7306 and 7308. Also, search engines and indexes 6902 are distributed among locations 7310 and 7312. Third party components/tools (described above) may also be local or distributed.

It is noted that the locations described in FIGS. 69-73 may be owned by and/or under the control of a single corporate entity, or owned and/or under the control of multiple corporate entities. Also, the locations depicted in FIGS. 69-73 may be local to one another, or may be very remote from one another, or may be combinations thereof.

The invention supports a number of economic models for using the system. For example, the system supports a pay-for-use service, where usage of the system is metered and billed accordingly. The invention also includes other economic models, such as but not limited to an unlimited service plan, where for a given fee the user can download any number of documents and have unlimited use of the system, and a hybrid plan where for a fee the user can download a given number of documents and use the system for a given amount of time or transactions (or some other metric). Once these limits are exceeded, the user pays on a per-use rate.

The invention also supports a model whereby meta-data version (v1) can be distributed to the client at time (t1). Later, an updated version of the meta-data (v2) can be distributed at time (t2). This is extremely valuable because certain aspects of the meta-data associated with a patent are mutable (e.g., assignee and U.S. classification). By providing these version numbers in the database, the host system can automatically provide meta-data updates to the remote system. Charges for these updates can be included within the user's existing economic model (above), or via a separate economic model.

The invention also supports electronic ordering of data objects (such as patents), and automatic downloading and installing of such data objects. In this embodiment, a user can electronically place an order for data objects, such as but not limited to via modem or over the Internet. An automated process takes the order, performs any necessary financial accounting and billing, retrieves the requested data items from a database, and downloads the retrieved data items/objects to the requestor via the Internet, for example. At the user computer, the received data objects are automatically decrypted (if necessary), unpacked (if necessary), unzipped (if necessary), installed in the proper directories, noted in the computer registry, and/or other tasks to automatically install the received data objects for use at the user site.

Assignee Processing

The invention supports processing related to corporate entities. For example, the invention includes functionality to identify and process patents assigned to a particular corporate entity.

To achieve effective corporate entity related processing, it is important that references to a given corporate entity in a data set be consistent throughout the data set. For example, if the user wishes to identify all patents assigned to Aurigin, Inc., then it is important that all references to Aurigin in the database utilize the same name, such as "Aurigin, Inc.". Processing conducted by the invention will not be as effective if different representations are used to represent a given corporate entity, such as "Origin" and "Aurigin, Incorporated" to represent Aurigin.

As noted above, the invention preferably receives patent data (herein called "raw data") from national patent offices. Such patent data often includes different representations of a given corporate entity. Accordingly, corporate entity related processing over such data is not entirely effective.

The invention solves this problem by processing assignee data contained in patent raw data. Such processing, called assignee name processing, is depicted in a flowchart 7402 of FIG. 74.

5 In step 7406, a normalized assignee name is selected for an entity. For example, the name "Aurigin" may be selected as a normalized assignee name for the corporate entity "Aurigin, Inc.".

10 In step 7408, the patent raw data is analyzed to identify all name representations for the corporate entity being considered. For example, in reviewing the data set, the following representations for Aurigin, Inc. may be identified:

Aurigin
Origin
Aurigin, Inc.
Aurigin, Incorporated
15 Origin, Inc.
Origin, Incorporated

20 Step 7408 may be performed manually, automatically, or a combination thereof. A variety of factors are considered to determine whether or not two representations refer to the same company or two independent and unrelated companies, such as but not limited to company name, company address, etc.

In step 7410, the normalized assignee name selected in step 7406 is linked to the name representations identified in step 7408.

25 Preferably, step 7410 is performed by replacing instances of the name representations appearing in database tables with the normalized assignee name. Preferably, the patent raw data is not modified.

The processing depicted in FIG. 74 is not limited to assignee name processing, but is instead applicable whenever there is a need that representations

of a given object be consistent throughout a data set. For example, with regard to patents, for example, the processing of FIG. 74 can be applied to inventor names, law firms, patent examiners, classes and subclasses (that change over time), any other bibliographic data, etc.

5 ***Input of Arbitrary and Potentially Diverse Data Objects***

10 The present invention enables the automatic or semi-automatic input into the system of data objects of any format and/or content. Such data objects are stored in databases of the system. The invention allows the display and annotation of the data objects. The data objects can also be searched. The
15 invention supports other processing of the data objects, as described herein, and as also described in "System, Method, and Computer Program Product for Managing and Analyzing Intellectual Property (IP) Related Transactions," Ser. No. 09/138,368, filed August 21, 1998; "Using Hyperbolic Trees to Visualize Data Generated By Patent-Centric and Group-Oriented Data Processing," Ser.
20 No. 08/921,369, filed August 29, 1997; and "System, Method, and Computer Program Product for Patent-Centric and Group-Oriented Data Processing," Ser. No. 08/867,392, filed June 2, 1997, all of which are incorporated herein by reference in their entirety.

25 In embodiments of the invention, information is extracted from the data objects. The extracted information is also stored. Preferably, the extracted information is stored in relational databases, data mining databases, or other types of databases. This is similar to storing bibliographic and other information from patents in relational databases, as described above, and as further described in U.S. Patent Application "System, Method and Computer Program Product for Patent-Centric and Group-Oriented Data Processing," Ser. No. 08/867,392, filed June 2, 1997, which is herein incorporated by reference in its entirety. The extracted data that is stored in the relational or other databases can be used for

sophisticated data mining and other processing related to the underlining data objects, as described herein.

Preferably, the format of data objects that are input into the system is at least partially regular and consistent. By this, it is meant that for a given data object type, a given piece of information (such as the document title) is always located at substantially the same position in data objects of the data object type. The invention makes use of the regular and consistent arrangement of data in data objects to extract data from the data objects. The extracted data is stored in relational databases, data mining databases, or other types of databases.

This operation of the invention is represented by a flow chart 8402 in FIG. 84. It is noted that the steps of flowchart 8402 are performed for a particular data object type, which shall herein be referred to as the "current data object type" for reference purposes.

In step 8404, the current data object type is analyzed to identify information of interest 8506 contained in data objects of the current data object type. The identified information of interest 8506 represents information that it is to be extracted from data objects of the current data object type. For example, if the current data object type is U.S. patents, then the information of interest 8506 identified in step 8404 may include the patent number, the inventors, the assignee, the cited references, the number of claims, the number of figures, etc.

The information of interest 8506 of the current data object type can be identified in step 8404 by a review of an example data object 8502 of the current data object type (see FIG. 85). Alternatively, the information of interest 8506 can be identified in step 8404 by other means. For example, a data object standard 8504 that characterizes/defines/describes the current data object type, and that specifies the data content and the locations of the data content in data objects of the current data object type, can be reviewed to identify the information of interest in step 8404.

In step 8406, the system identifies where the information of interest 8506 (identified in step 8404) is located in data objects of the current data object type. Again, step 8406 can be performed by a review of an example data object 8502 of the current data object, or by review of a data object standard 8504 that characterizes the current data object type.

In step 8408, one or more new database tables are created, if necessary, to store the data that it to be extracted from data objects of the current data object type. It is noted that step 8408 is optional, in that database tables for storing the extracted information may already exist.

In step 8410, one or more database tables 8510 (one or more of which may have been created in 8408) are selected. The selected database tables 8510 will store the information extracted from data objects of the current data object type. The database tables 8510 selected in step 8410 are represented, for example, as relational database tables 8510 in FIG. 85.

In step 8412, the columns (fields) in the selected database tables 8510 where the information extracted from the data objects of the current data object type is to be stored are selected. This is represented in FIG. 85, for example, where it is specified that data in location 8508A is stored in column 8512 of relational database 8510. Similarly, data in locations 8508B, 8508C and 8508D of the data objects of the current data object type are stored in columns 8514, 8516 and 8518, respectively, of the relational database tables 8510.

After completion of step 8412, the system is prepared to receive and extract data from data objects of the current data object type. Accordingly, in step 8414, a data object of the current data object type is received.

In step 8416, the system determines the type of the received data object. In particular, the system in step 8416 determines that the received data object is of the current data object type. The system can determine the type by reviewing information that accompanied the data object when it was received in step 8414. In particular, in step 8414, an indication of the type of the data object may also

have been provided to the system. Alternatively, the system can analyze the received data object to determine its type.

In step 8418, the system extracts information from the received data object in accordance with the data object's type. Preferably, this extraction operation does not change or otherwise affect the received data object. The system knows where to extract information of interest 8506 by reference to the locations 8508 identified in step 8406.

In step 8420, the system stores the extracted information in columns of the database tables 8510 associated with the current database type, as determined in steps 8410 and 8412.

In step 8422, the system stores the data object itself.

Tool Box Embodiments

A variety of widgets (components) of the invention are described herein. The invention is directed to tool box embodiments where any number of these widgets are packaged as separately accessible components that can be added to other applications, or used to customize the system described herein. For example, and without limitation, an extension of the user interface includes the ability to "mix and match" specific widgets or components associated with the user interface. Consider FIGS. 16 and 75, and the "kind pane" (7504). These could be packaged as separately accessible components of the API and made available as a "toolkit". The same is true with all other components of the invention. Users could build their own user interface or other applications -- perhaps by embedding the context browser into other applications, such as a spreadsheet in Microsoft Excel.

Example Implementation

In an embodiment of the present invention, the system and components of the present invention described herein are implemented using well known computers, such as a computer 8602 shown in FIG. 86. The computer 8602 can be any commercially available and well known computer capable of performing the functions described herein, such as computers available from International Business Machines, Apple, Silicon Graphics Inc., Sun, HP, Dell, Compaq, Digital, Cray, etc.

The computer 8602 includes one or more processors (also called central processing units, or CPUs), such as a processor 8606. The processor 8606 is connected to a communication bus 8604. The computer 8602 also includes a main or primary memory 8608, preferably random access memory (RAM). The primary memory 8608 has stored therein control logic (computer software), and data.

The computer 8602 also includes one or more secondary storage devices 8610. The secondary storage devices 8610 include, for example, a hard disk drive 8612 and/or a removable storage device or drive 8614. The removable storage drive 8614 represents a floppy disk drive, a magnetic tape drive, a compact disk drive, an optical storage device, tape backup, ZIP drive, JAZZ drive, etc.

The removable storage drive 8614 interacts with a removable storage unit 8616. As will be appreciated, the removable storage unit 8616 includes a computer usable or readable storage medium having stored therein computer software (control logic) and/or data. The removable storage drive 8614 reads from and/or writes to the removable storage unit 8616 in a well known manner.

Removable storage unit 8616, also called a program storage device or a computer program product, represents a floppy disk, magnetic tape, compact disk, optical storage disk, ZIP disk, JAZZ disk/tape, or any other computer data storage device. Program storage devices or computer program products also include any device in which computer programs can be stored, such as hard drives, ROM or memory cards, etc.

In an embodiment, the present invention is directed to computer program products or program storage devices having software that enables the computer 8602 to perform any combination of the functions described herein.

5 Computer programs (also called computer control logic) are stored in main memory 8608 and/or the secondary storage devices 8610. Such computer programs, when executed, enable the computer 8602 to perform the functions of the present invention as discussed herein. In particular, the computer programs, when executed, enable the processor 8606 to perform the functions of the present invention. Accordingly, such computer programs represent controllers of the
10 computer 8602.

The computer 8602 also input/output/display devices 8622, such as monitors, keyboards, pointing devices, etc.

The computer 8602 further includes a communication or network interface 8618. The network interface 8618 enables the computer 8602 to communicate
15 with remote devices. For example, the network interface 8618 allows the computer 8602 to communicate over communication networks, such as LANs, WANs, the Internet, etc. The network interface 8618 may interface with remote sites or networks via wired or wireless connections. The computer 8602 receives data and/or computer programs 8620 via the network interface 8618. The
20 electrical/magnetic signals having contained therein data and/or computer programs 8620 received or transmitted by the computer 8602 via interface 8618 also represent computer program product(s).

The invention can work with software, hardware, and operating system implementations other than those described herein. Any software, hardware, and
25 operating system implementations suitable for performing the functions described herein can be used.

Conclusion

While various embodiments of the present invention have been described above, it should be understood that they have been presented by way of example only, and not limitation. Thus, the breadth and scope of the present invention should not be limited by any of the above-described exemplary embodiments, but should be defined only in accordance with the following claims and their equivalents.

What Is Claimed Is:

1. A computer implemented method of enabling review of patents in an electronic patent shoe, comprising the steps of:

5 (1) causing generation of an electronic patent shoe comprising a plurality of patents;

(2) causing display of a list of patents in said electronic patent shoe;

(3) causing display of image data representative of at least a portion of a patent from said electronic patent shoe;

10 (4) causing, pursuant to a command to view a next patent, retrieval and display of at least image data representative of at least a portion of said next patent; and

(5) causing, pursuant to a command to view a previous patent, retrieval and display of at least image data representative of at least a portion of said previous patent.

15 2. The method of claim 1, further comprising the steps of:

(6) determining whether requested information pertaining to a patent is locally available; and

(7) if said requested information is not locally available, then providing an option to preview said requested information; and

20 (8) if said option is selected, then retrieving and displaying at least a portion of said requested information.

3. A method of context data processing, comprising the steps of:

(1) selecting one or more contexts each including one or more attributes and a plurality of data objects that satisfy said attributes;

25 (2) displaying a list of data objects contained in said contexts; and

(3) processing a plurality of said data objects contained in said contexts.

4. The method of claim 3, wherein said contexts are groups.

5. The method of claim 3, wherein each of said contexts is associated with a data object type and comprises data objects of said data object type.

6. The method of claim 3, wherein said data objects are of interest to a corporate entity.

7. The method of claim 6, wherein said data objects are patents, and wherein step (3) comprises the step of:

(a) performing patent-centric processing of said data objects.

8. The method of claim 7, wherein step (a) comprises the step of:
generating a patent citation tree comprising patents and non-patent data objects that (i) are cited by a base patent, or (ii) cite a base patent.

9. The method of claim 8, further comprising the step of:
visualizing said patent citation tree as a hyperbolic tree.

10. The method of claim 9, wherein each node of said hyperbolic tree comprises state information including display attribute information.

11. The method of claim 7, wherein step (a) comprises the step of:
generating a patent claim tree comprising nodes corresponding to claims in one of said patents, and links between said nodes indicative of dependencies among said claims.

12. The method of claim 11, further comprising the step of:
displaying at least a portion of a claim corresponding to one of said nodes
when a selection device is positioned proximate to said one of said nodes.

5 13. The method of claim 11, further comprising the step of:
displaying a dependent claim, a base claim of said dependent claim, and
any intervening claims between said dependent claim and said base claim upon
appropriate user command.

10 14. The method of claim 3, further comprising the steps of:
(4) generating a data object family tree comprising a plurality of data
objects and indicative of relationships between said data objects; and
(5) displaying said data object family tree.

15. The method of claim 14, wherein step (5) comprises the step of:
displaying said data object family tree as a hyperbolic tree.

15 16. The method of claim 3, further comprising the steps of:
(4) generating at least one annotation.

17. The method of claim 16, wherein step (4) comprises the step of:
generating a document annotation that is linked to at least a portion of at
least one of said data objects, and that has a scope of a group containing said at
least one of said data objects.

20 18. The method of claim 17, wherein said document annotation is linked to at
least a portion of image data related to said at least one of said data objects.

19. The method of claim 16, wherein step (4) comprises the step of:

generating a group annotation that has a scope of a group active when said group annotation was created.

20. The method of claim 16, wherein step (4) comprises the step of:
generating a data object type annotation that has a scope of a data object
5 type active when said data object type annotation was created.

21. The method of claim 16, wherein step (4) comprises the step of:
generating a case annotation that is linked to at least a portion of at least
one of said data objects, and that has a scope of a case active when said case
annotation was created.

10 22. The method of claim 16, wherein step (4) comprises the step of:
generating an enterprise annotation that is linked to at least a portion of
at least one of said data objects, and that is always accessible to users with
appropriate authorization.

15 23. A system for processing data, comprising:
an intellectual property asset manager (IPAM);
a plug-in manager coupled to said IPAM; and
at least one plug-in coupled to said plug-in manager.

24. The system of claim 23, further comprising:
at least one external data processing component coupled to said plug-in.

20 25. The system of claim 24, wherein said external data processing component
displays data using at least graphs.

26. The system of claim 24, wherein said external data processing component displays data using at least maps.

27. The system of claim 24, wherein said plug-in manager has a first application programming interface (API), and said external data processing component has a second API.

28. The system of claim 27, wherein said plug-in comprises:
means for translating messages from said plug-in manager to said external data processing component to a format conforming to said second API; and
means for translating messages from said external data processing component to said plug-in manager to a format conforming to said first API.

29. A method of assignee name processing, comprising the steps of:
(1) selecting a normalized assignee name for an entity;
(2) identifying name representations of said entity in a data set; and
(3) linking said name representations to said normalized assignee name.

30. The method of claim 29, wherein step (3) comprises the steps of:
substituting at least some instances of said name representations with said normalized assignee name.

31. A method of context data processing, comprising the steps of:
(1) accessing databases of patents and patent bibliographic information;
(2) accessing one or more contexts, each of said one or more contexts comprising any number of patents represented in said databases; and

(3) automatically performing patent centric and context oriented processing of at least some of said patents in at least one of said one or more contexts.

32. The method of claim 31, further comprising the step of:

(4) annotating a patent represented in said databases.

33. The method of claim 32, wherein step (4) comprises the step of: annotating said patent using an annotation form.

34. The method of claim 31, wherein step (3) comprises: generating and displaying data object families.

35. A system for processing data, comprising:
an intellectual property asset manager (IPAM); and
at least one external data processing component coupled to said IPAM to expand functionality of said IPAM, wherein said external data processing component displays data from said IPAM using at least one of graphs and maps.

36. The system of claim 35, further comprising:
means for accessing databases of patents and patent bibliographic information;

means for accessing one or more groups, each of said one or more groups comprising any number of said patents represented in said databases; and

means for automatically performing patent centric and group oriented processing of at least some of said patents in at least one of said one or more groups.

37. The system of claim 35, further comprising:
means for generating and displaying data object families.

701
(console)
jsm019 dmjg

Fig. 1

The image is a high-contrast, black and white scan of a computer monitor displaying a document management software interface. The main window is titled 'Auriga Foundation Server ds server 2 0000 - administ ator'. It features a table with columns for 'Document ID', 'Document', and 'Assignee'. The table lists numerous documents, many of which are from 'THE PROCTER & GAMBLE COMPANY' and 'PZIFER INC.'. To the left of the table is a sidebar with a tree view showing the application's structure, including folders like '171 docs', 'epi docs', 'group', and 'server'. Above the table, there are navigation icons and a search bar. To the right of the table, there is a 'Data Object Panel' showing a diagram of a ship. The screen is heavily annotated with handwritten notes and markings, including '105', '107', '104', and 'Data Object Panel'. There are also some printed annotations like 'Annotations pane' and 'Contents pane 106'.

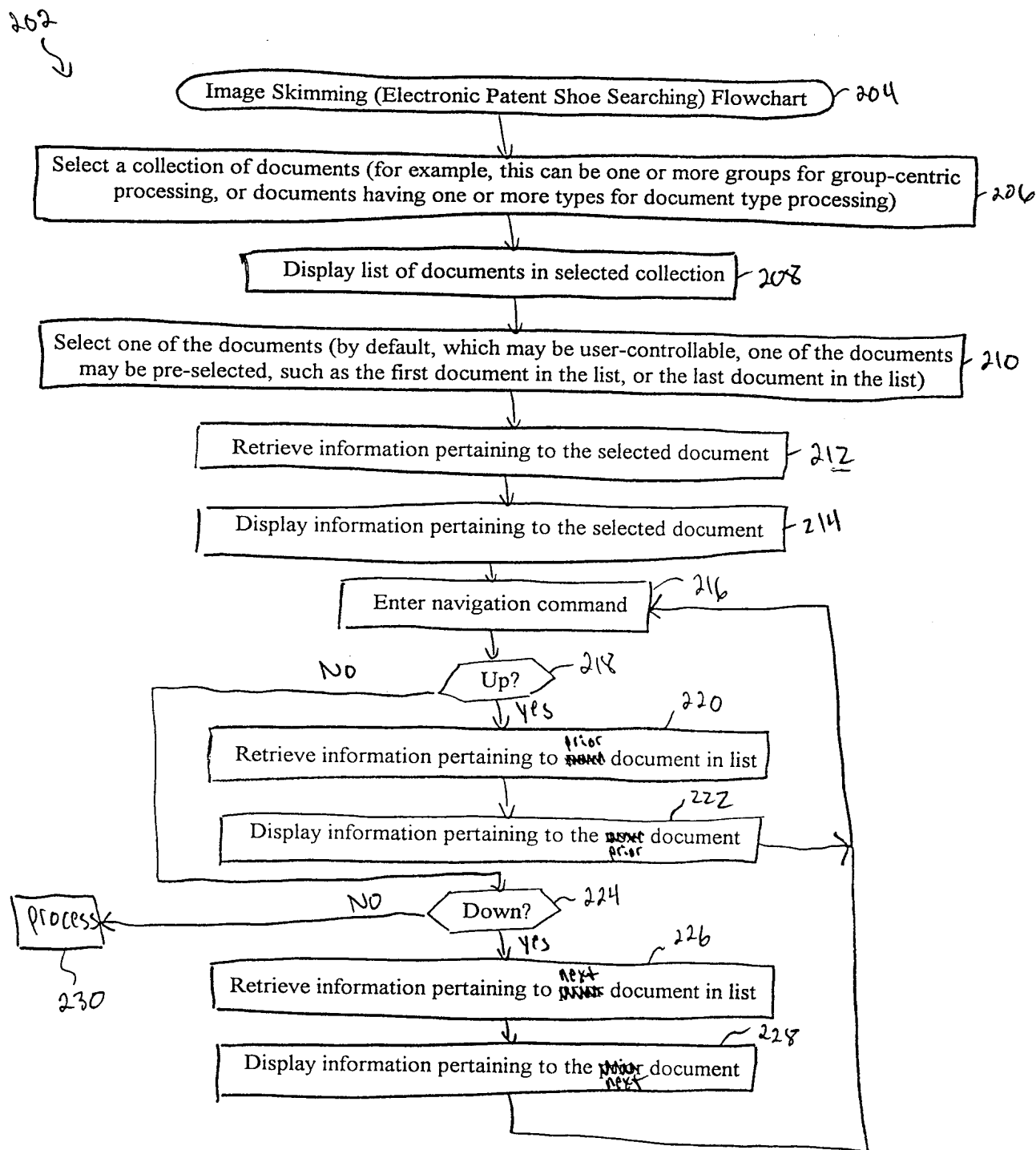
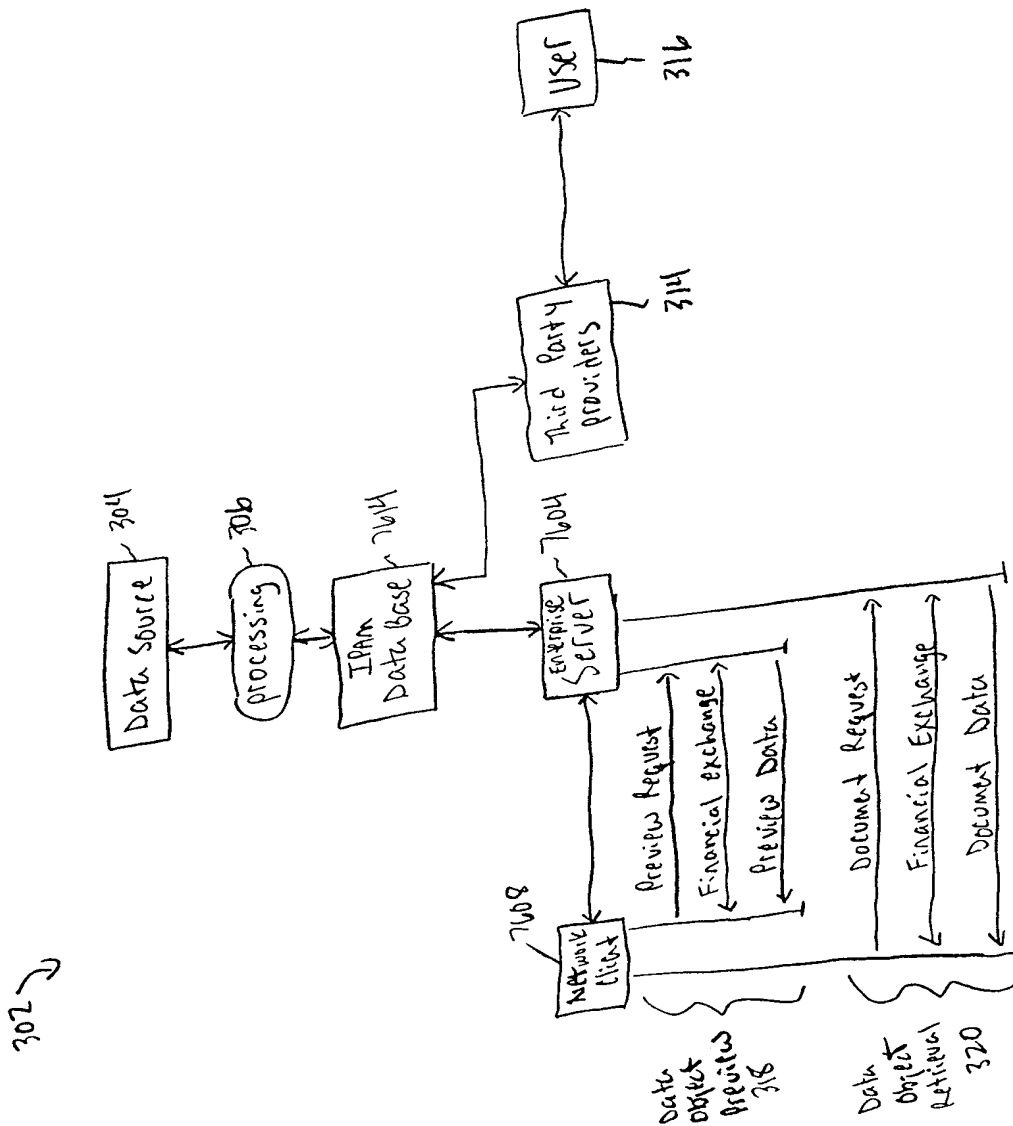


FIG. 2

3/99



4/99

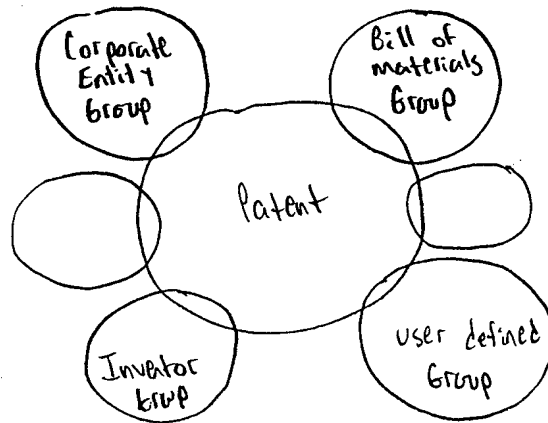


FIG. 4

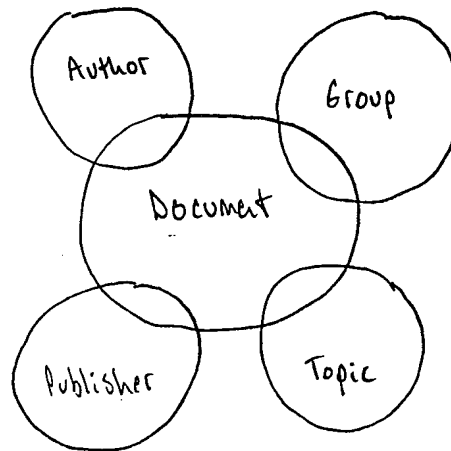
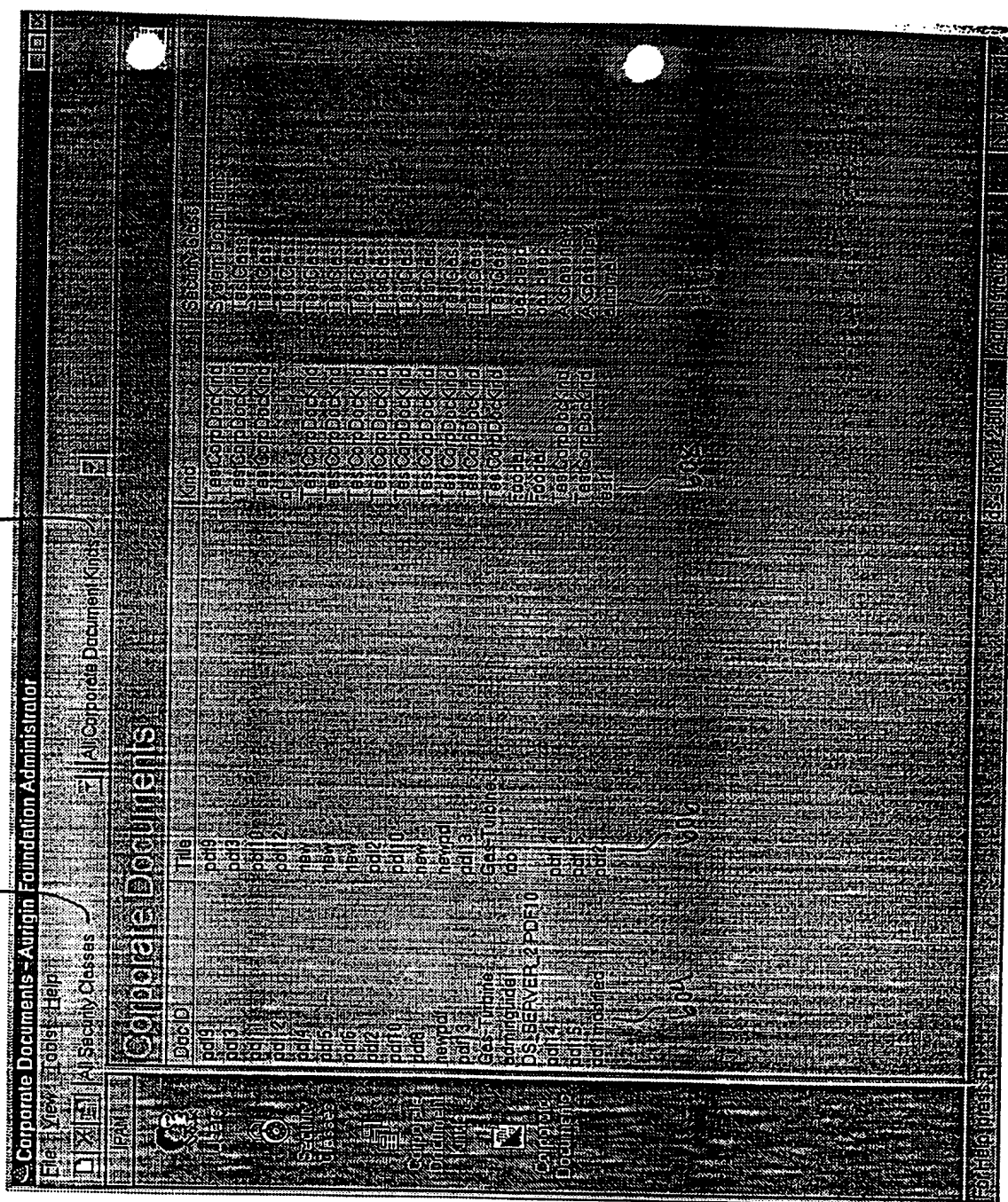
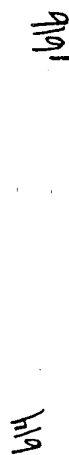
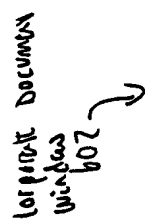
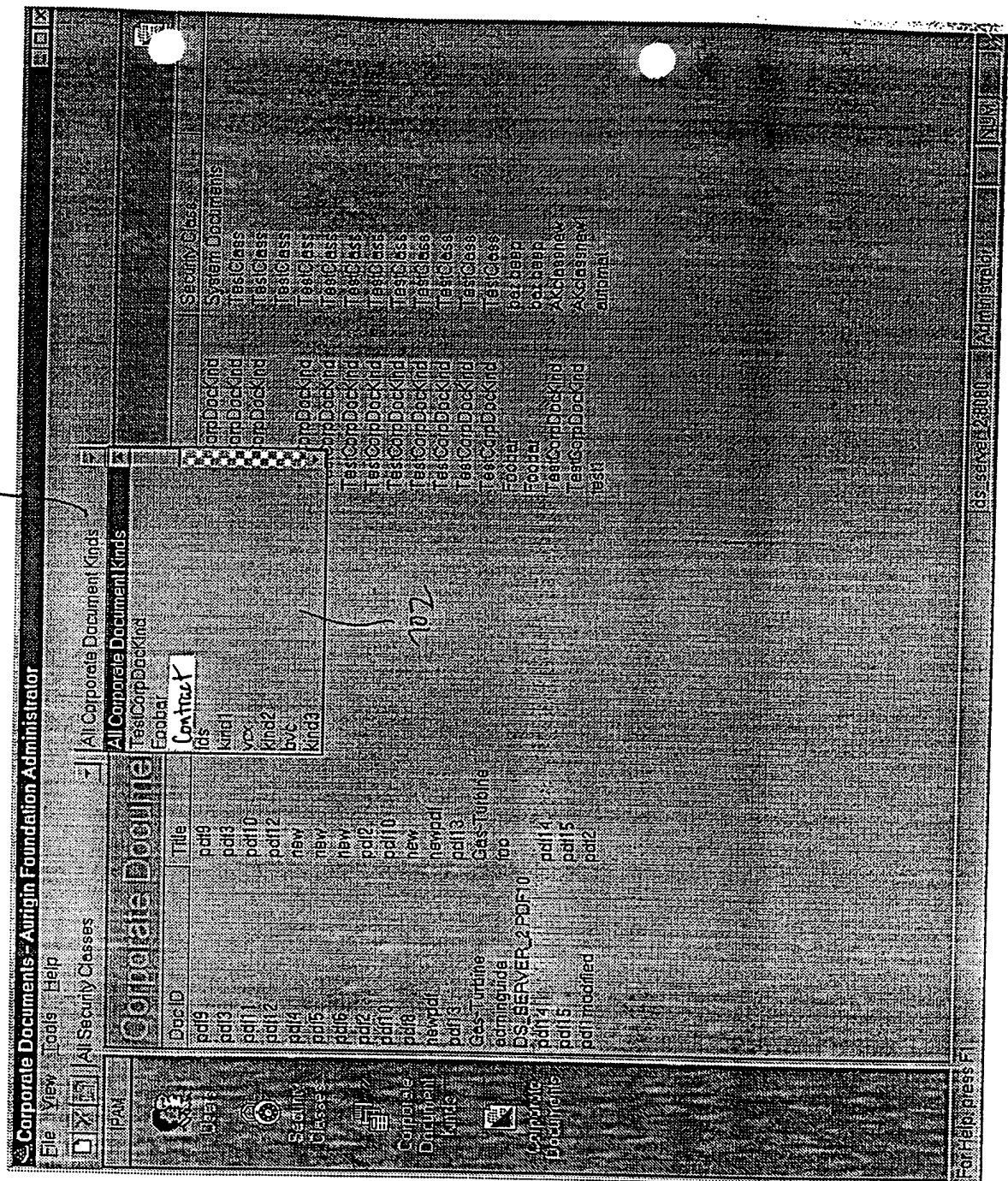


FIG. 5



Corporate Document
Wardens vol 2

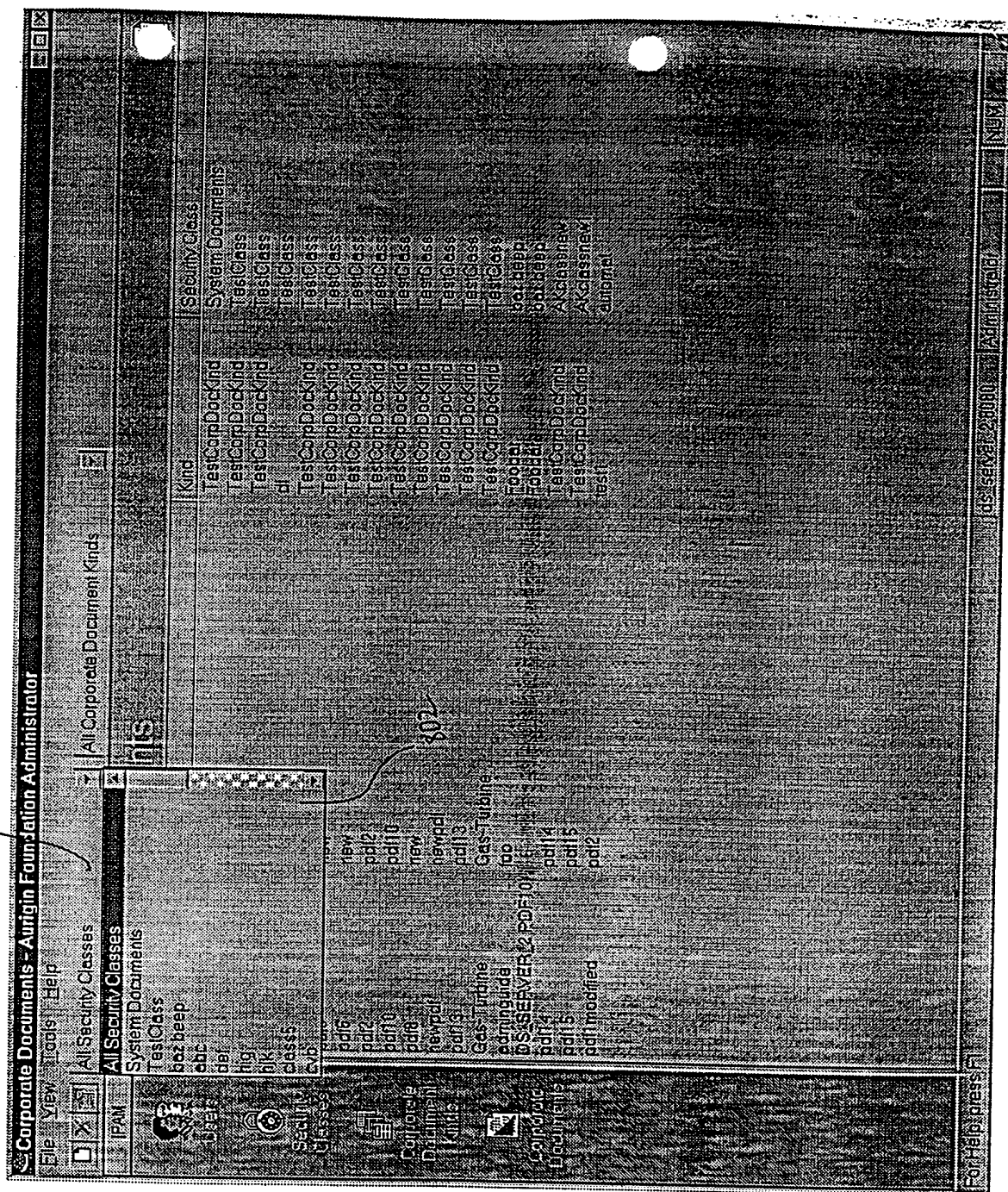
Fig 7



berpakte Document
wunders boer

519

Fig. 8



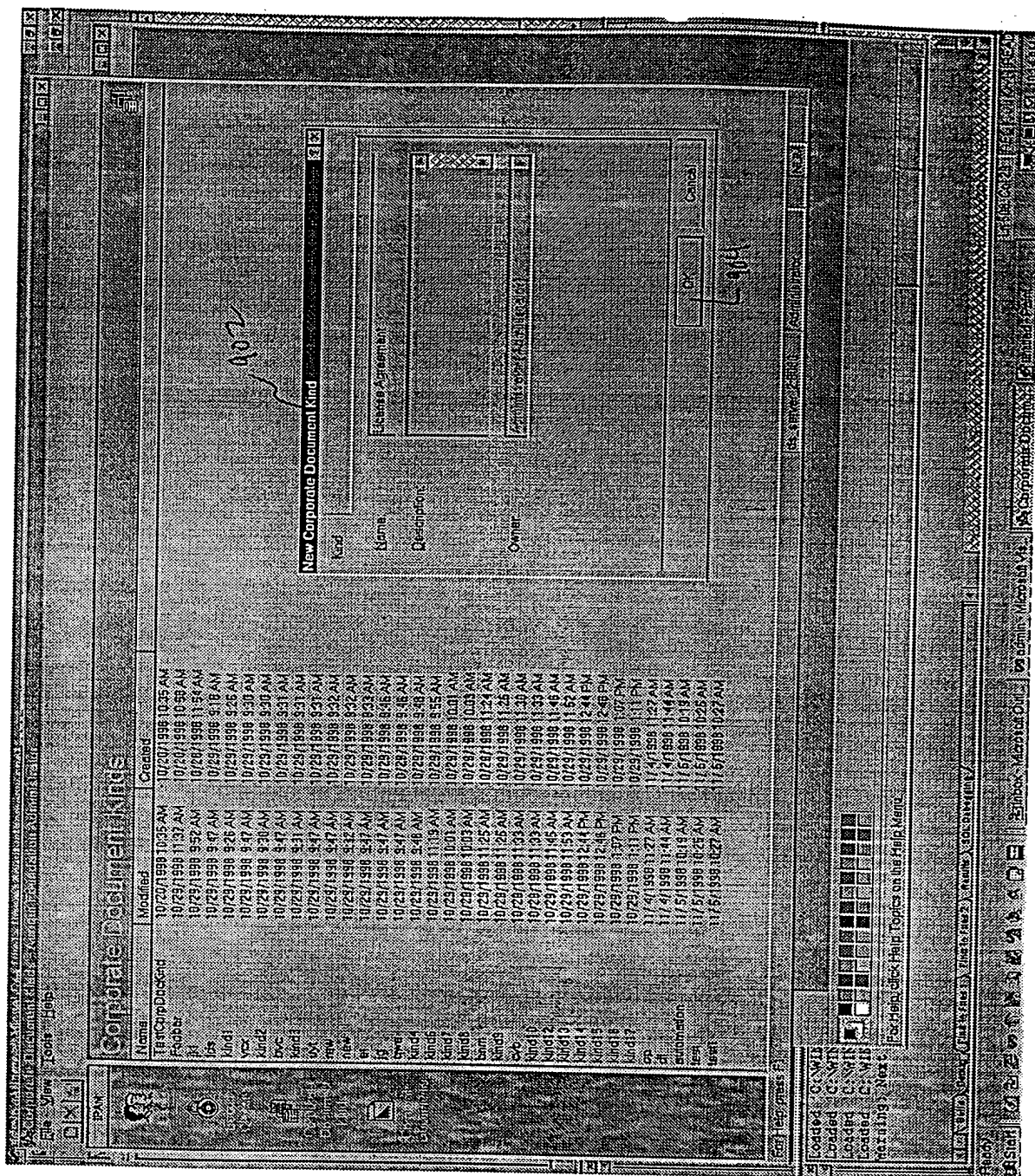


Fig. 9

10/99

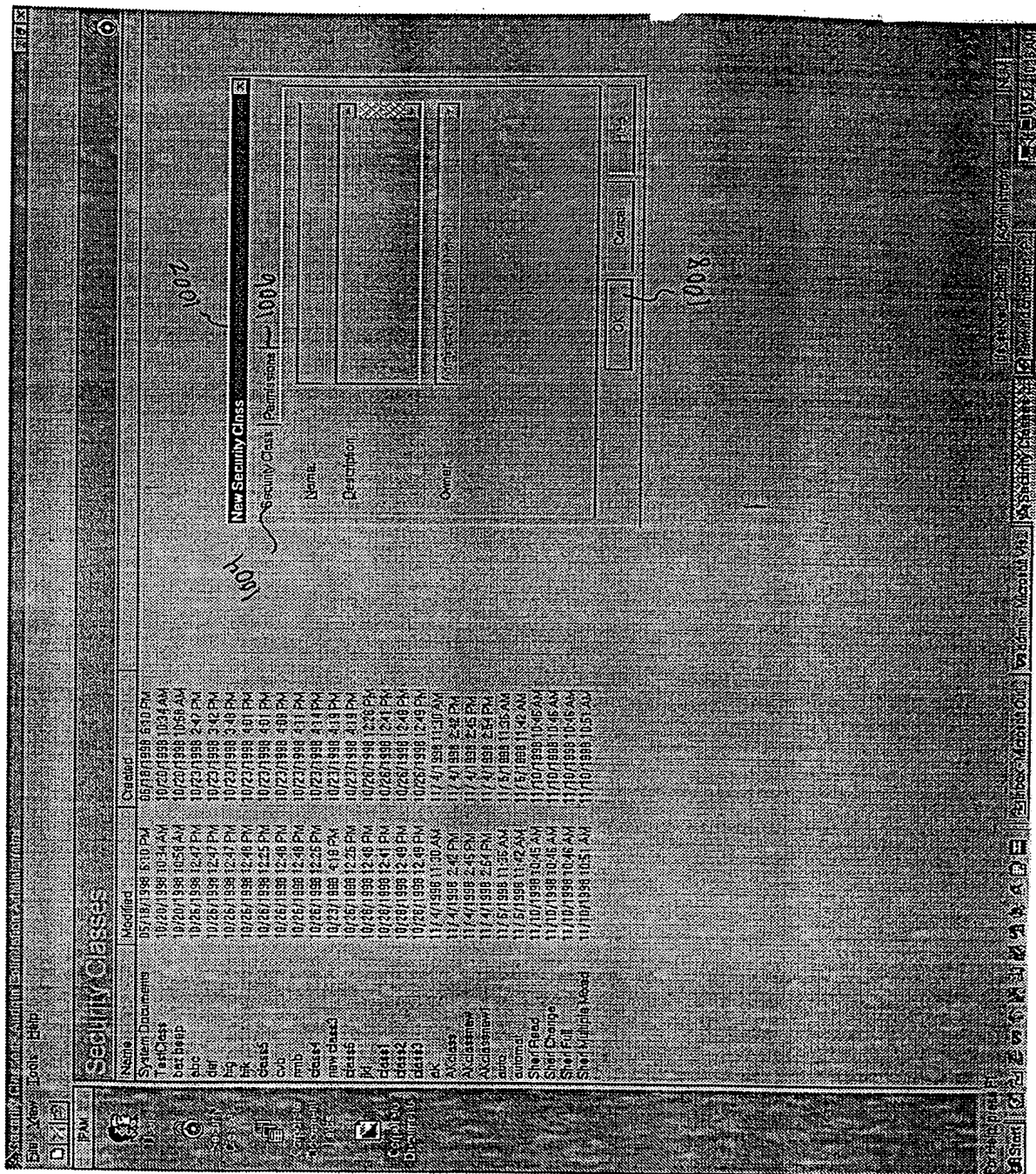


Fig. 10

11/99

New Security Class

Security Class: Admissions
Owner: newclass
Name:

permissions (inherit)
permissions (inherit)
users (Admin)

Type of Access:
 Add Remove

OK Cancel

1002

1008

Name	Modified	Created
System Documents	05/18/1998 8:10 PM	05/18/1998 8:10 PM
TstClass	10/20/1998 10:34 AM	10/20/1998 10:34 AM
bat help	10/20/1998 10:58 AM	10/20/1998 10:58 AM
dnc	10/26/1998 12:41 PM	10/23/1998 2:47 PM
dir	10/26/1998 12:41 PM	10/23/1998 3:42 PM
log	10/26/1998 12:41 PM	10/23/1998 3:40 PM
pk	10/26/1998 12:48 PM	10/23/1998 4:51 PM
class5	10/26/1998 12:25 PM	10/23/1998 4:51 PM
cob	10/26/1998 12:48 PM	10/23/1998 4:59 PM
rmb	10/26/1998 12:48 PM	10/23/1998 4:11 PM
class4	10/26/1998 12:25 PM	10/23/1998 4:14 PM
newClass5	10/23/1998 4:19 PM	10/23/1998 4:19 PM
class5	10/26/1998 12:25 PM	10/23/1998 4:18 PM
pk	10/26/1998 12:48 PM	10/26/1998 12:28 PM
class1	10/26/1998 12:41 PM	10/26/1998 12:41 PM
class2	10/26/1998 12:48 PM	10/26/1998 12:48 PM
class3	10/26/1998 12:49 PM	10/26/1998 12:48 PM
pk	11/7/1998 11:30 AM	11/7/1998 11:30 AM
A/class	11/7/1998 2:45 PM	11/7/1998 2:42 PM
A/classnew	11/7/1998 2:45 PM	11/7/1998 2:45 PM
auto	11/7/1998 2:54 PM	11/7/1998 2:54 PM
admsn	11/7/1998 11:36 AM	11/7/1998 11:36 AM
Share Read	11/7/1998 11:42 AM	11/7/1998 11:42 AM
Share Change	11/10/1998 10:45 AM	11/10/1998 10:45 AM
Share Full	11/10/1998 10:46 AM	11/10/1998 10:46 AM
Share Multiple Read	11/10/1998 10:51 AM	11/10/1998 10:51 AM

FILE //

12/99

1202

1210

1206

1216

1218

UNC File Path	1204	Browse...
Document ID	1208	
Title	1210	
Document Kind	1212	New
Storage Class	1214	New...

OK Cancel

FIG. 12

13/99

Corporate Document
Windows 1302

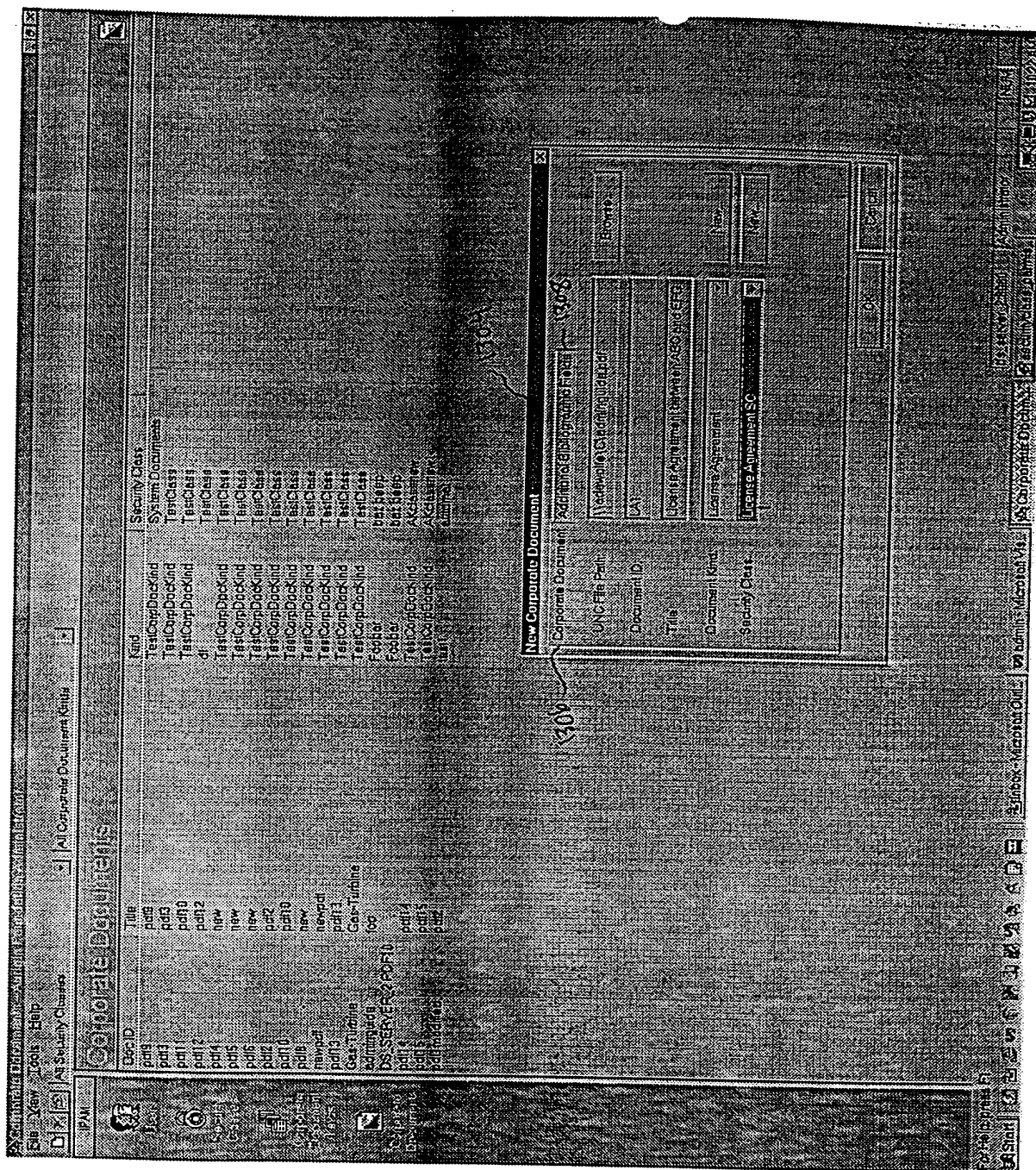


FIG. 13

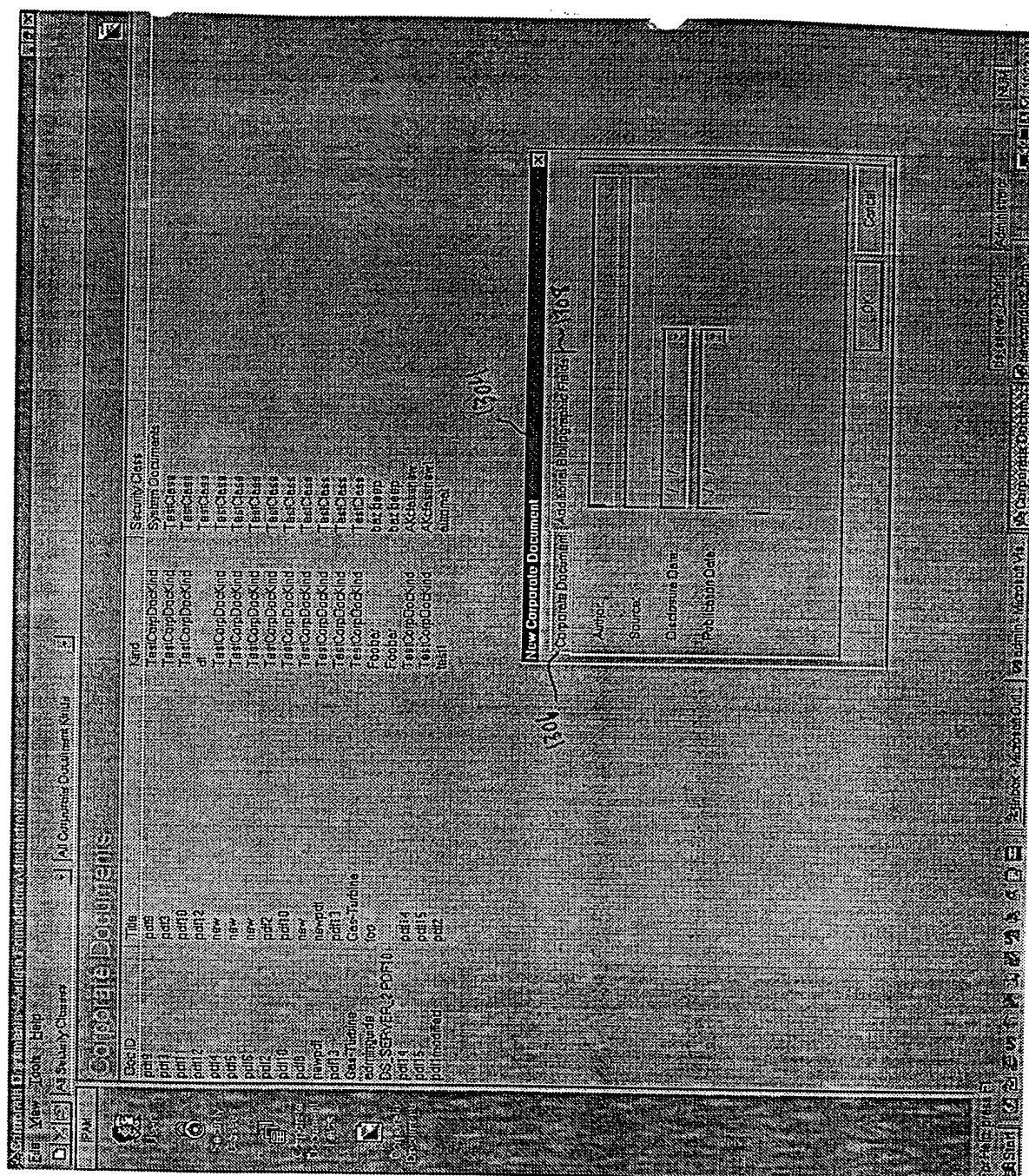


Fig. 14

Corporate Document
Window 1302

15/99

Corporate Downmix
Windows 1302 →

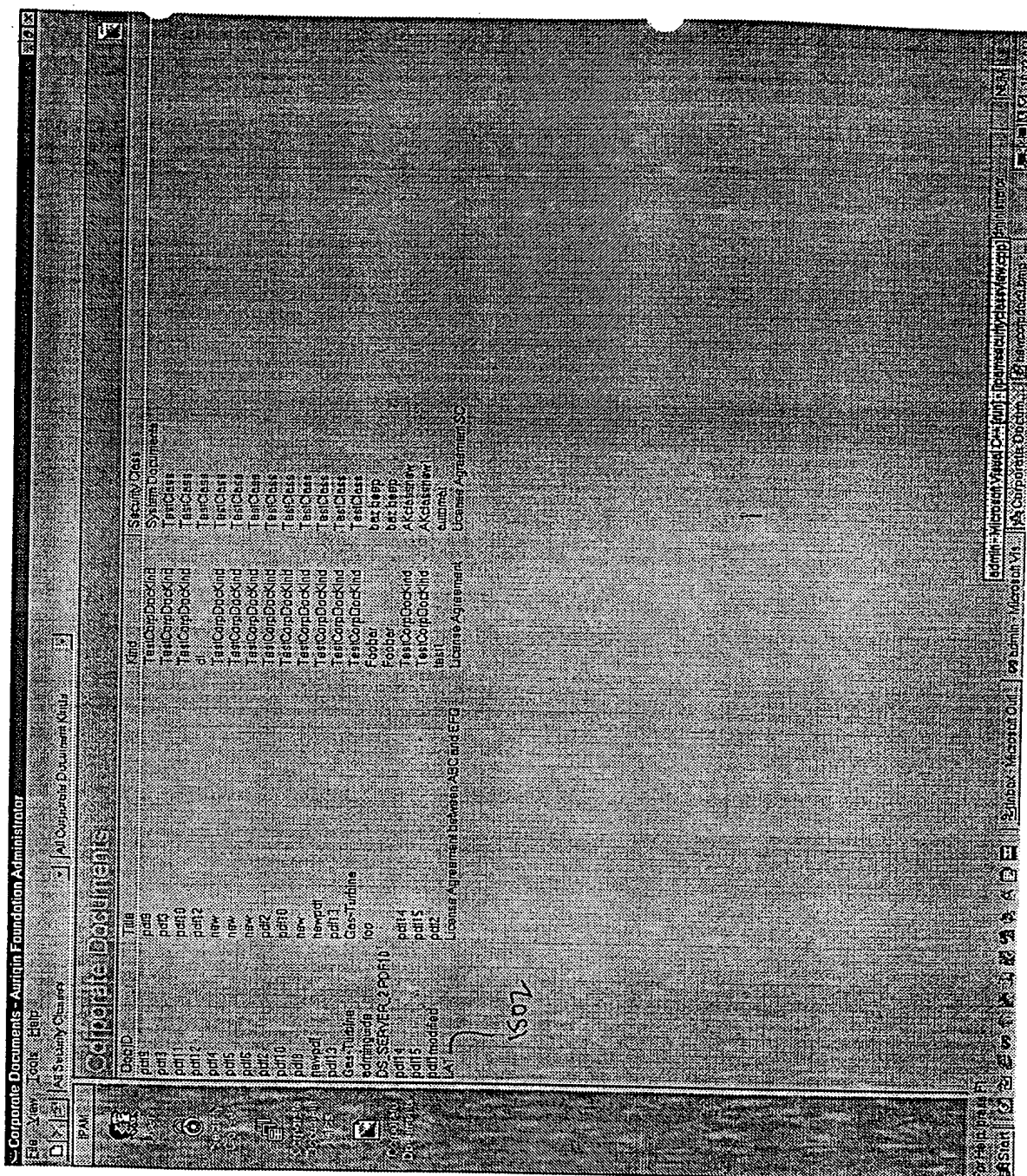
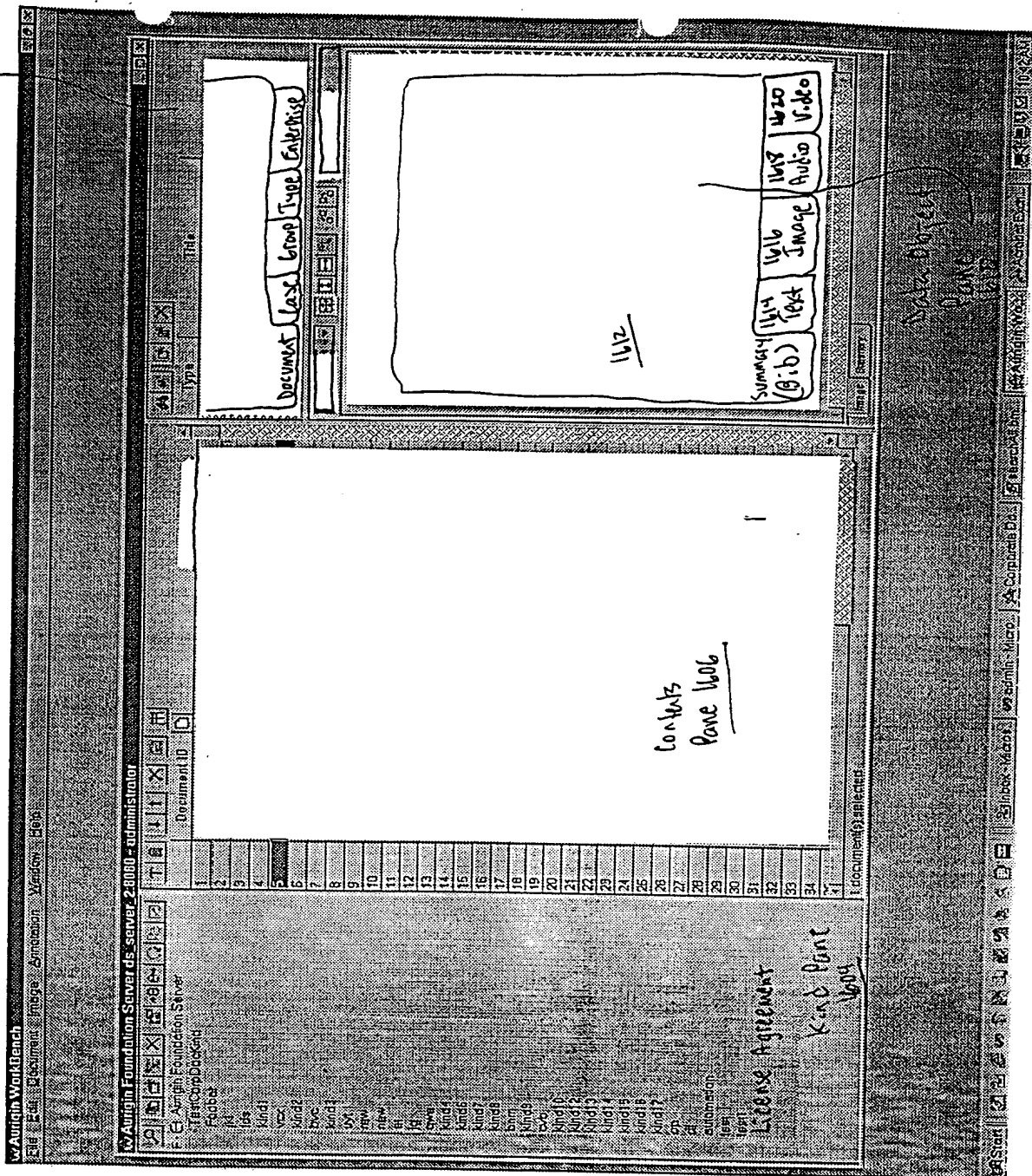


Fig. 15

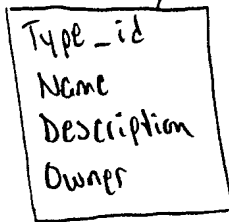
Context Browser 1662 →

Amphioxus Line 1108



17/99

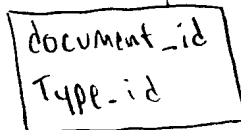
Type Table 1702



Type_id
Name
Description
Owner

FIG. 17

Type Document Xref 1802



document_id
Type_id

FIG. 18

18/99

Type-id	Name	...
	•	
	•	
T1	Automation	
T2	License Agreement	
	•	
	•	
	•	

Type table 1902

FIG. 19

document-id	Type-id	...
Contract 1	T1	
Contract 1	T2	
Contract 2	T2	
Contract 3	T1	
	•	
	•	
	•	

Type Document Xref Table 2002

FIG. 20

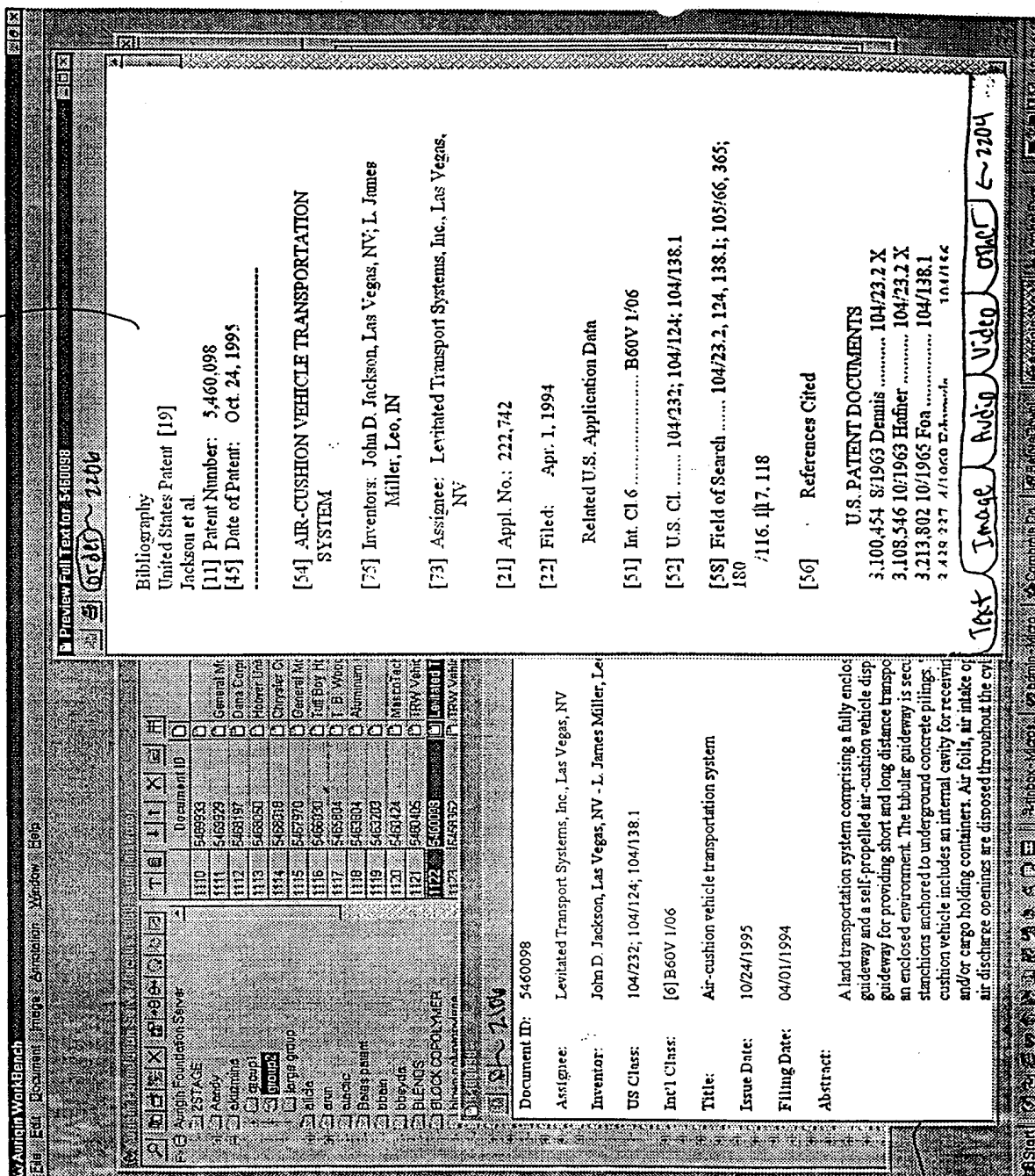
contacts page 2108

[illegible]

6509/Kind
pane
2/10/6

Fig. 2

Data object -
Page 704



226

Date _____
Object _____
Page 204

21/99

2302
↓

Annotation Type	Attached to a Data Object?	Scope of Annotation			
		Data Object	Case	Group	Type
Document	Yes			X	
Case	Yes		X		
Group	No			X	
Type	No				X
Enterprise	Yes	X			

FIG. 23

browser 2402 ↗

Fig. 24

Assign		Type		Title	
1	EP 0 751 255 A1	<input type="checkbox"/>	<input type="checkbox"/>		
2	EP 0 658 650 A1	<input type="checkbox"/>	<input type="checkbox"/>		
3	EP 0 353 402 A2	<input type="checkbox"/>	<input type="checkbox"/>		
4	EP 0 508 656 B1	<input type="checkbox"/>	<input type="checkbox"/>		
5	EP 0 508 650 B1	<input type="checkbox"/>	<input type="checkbox"/>		
6	EP 0 742 41	<input type="checkbox"/>	<input type="checkbox"/>		
7	5568742	<input checked="" type="checkbox"/>	<input type="checkbox"/>		
8	5584597	<input type="checkbox"/>	<input type="checkbox"/>		
9	5607859	<input type="checkbox"/>	<input type="checkbox"/>		
10	5629155	<input type="checkbox"/>	<input type="checkbox"/>		
11	5653767	<input type="checkbox"/>	<input type="checkbox"/>		
12	5655689	<input type="checkbox"/>	<input type="checkbox"/>		
13	5367459	<input type="checkbox"/>	<input type="checkbox"/>		
14	4722410	<input type="checkbox"/>	<input type="checkbox"/>		
15	4145107	<input type="checkbox"/>	<input type="checkbox"/>		
16	3785592	<input type="checkbox"/>	<input type="checkbox"/>		
17	3740704	<input type="checkbox"/>	<input type="checkbox"/>		
18	W0 3710017	<input type="checkbox"/>	<input type="checkbox"/>		
19	W0 3710017	<input type="checkbox"/>	<input type="checkbox"/>		
20	W0 3710017	<input type="checkbox"/>	<input type="checkbox"/>		
21	W0 3710017	<input type="checkbox"/>	<input type="checkbox"/>		
22	W0 3710017	<input type="checkbox"/>	<input type="checkbox"/>		

Document ID	Assign
EP 0 751 255 A1	YOUTH SUZUKI PAPER
EP 0 658 650 A1	WESTINGHOUSE ELECT
EP 0 353 402 A2	FORESE, FRANCESCO
EP 0 508 656 B1	NEURODEX FRANCE
EP 0 508 650 B1	WESTINGHOUSE ELECT
EP 0 742 41	Town, Semiconductor Me
5568742	Komatsu Ltd
5584597	Shinko Cellular Mts (di
5607859	Kaiseng Engineering, Inc
5629155	Nisori Corporation
5653767	Kobayashi Katsuo Komatsu
5655689	Enjin Siki GmHh Onaka Ei
5367459	Cellular Industrial Inc
4722410	Cellular Industrial Inc
4145107	Sperry Rand Corporation
3785592	IN-ONT, INC
3740704	WESSEL-KENTON, JELLY
W0 3710017	PCR SCIENTIFIC, INC
W0 3710017	PCR SCIENTIFIC, INC
W0 3710017	PCR SCIENTIFIC, INC
W0 3710017	PCR SCIENTIFIC, INC

Document ID	Assign
W0 3710017	PCR SCIENTIFIC, INC
W0 3710017	PCR SCIENTIFIC, INC
W0 3710017	PCR SCIENTIFIC, INC
W0 3710017	PCR SCIENTIFIC, INC

Document ID	Assign
W0 3710017	PCR SCIENTIFIC, INC
W0 3710017	PCR SCIENTIFIC, INC
W0 3710017	PCR SCIENTIFIC, INC
W0 3710017	PCR SCIENTIFIC, INC

Document ID	Assign
W0 3710017	PCR SCIENTIFIC, INC
W0 3710017	PCR SCIENTIFIC, INC
W0 3710017	PCR SCIENTIFIC, INC
W0 3710017	PCR SCIENTIFIC, INC

Document ID	Assign
W0 3710017	PCR SCIENTIFIC, INC
W0 3710017	PCR SCIENTIFIC, INC
W0 3710017	PCR SCIENTIFIC, INC
W0 3710017	PCR SCIENTIFIC, INC

Document ID	Assign
W0 3710017	PCR SCIENTIFIC, INC
W0 3710017	PCR SCIENTIFIC, INC
W0 3710017	PCR SCIENTIFIC, INC
W0 3710017	PCR SCIENTIFIC, INC

Document ID	Assign
W0 3710017	PCR SCIENTIFIC, INC
W0 3710017	PCR SCIENTIFIC, INC
W0 3710017	PCR SCIENTIFIC, INC
W0 3710017	PCR SCIENTIFIC, INC

Document ID	Assign
W0 3710017	PCR SCIENTIFIC, INC
W0 3710017	PCR SCIENTIFIC, INC
W0 3710017	PCR SCIENTIFIC, INC
W0 3710017	PCR SCIENTIFIC, INC

Document ID	Assign
W0 3710017	PCR SCIENTIFIC, INC
W0 3710017	PCR SCIENTIFIC, INC
W0 3710017	PCR SCIENTIFIC, INC
W0 3710017	PCR SCIENTIFIC, INC

Document ID	Assign
W0 3710017	PCR SCIENTIFIC, INC
W0 3710017	PCR SCIENTIFIC, INC
W0 3710017	PCR SCIENTIFIC, INC
W0 3710017	PCR SCIENTIFIC, INC

Document ID	Assign
W0 3710017	PCR SCIENTIFIC, INC
W0 3710017	PCR SCIENTIFIC, INC
W0 3710017	PCR SCIENTIFIC, INC
W0 3710017	PCR SCIENTIFIC, INC

Document ID	Assign

23/99

Browser 2402

Text Window 2502

WAutigwin Workbench File Edit Document Image Annotation Window Help

Unit States Patent (19)

Murayama

[11] Patent Number: 5,888,742

[45] Date of Patent: Sep. 16, 1997

[54] APPARATUS FOR DETERMINING POSITION OF

[75] Inventor: Osamu Murayama, Isehara, Japan

[73] Assignee: Komatsu Ltd., Japan

[21] Appl. No.: 666,329

[22] Filed: May 28, 1998

[22] PCT Filed: Dec. 8, 1994

[66] PCT No.: PCT/JP94/02044

\$ 371 Date: May 28, 1998

\$ 102(e) Date: May 28, 1998

[87] PCT Pub. No.: WO95/16184

PCT Pub. Date: Jun. 15, 1995

Related U.S. Application Data

[30] Foreign Application Priority Data

Dec. 7, 1993 Japan 5-308702

Dec. 14, 1993 Japan 5-313609

[51] Int. Cl. G01B 11/14

[52] U.S. Cl. 364/559; 318/560; 356/375; 701/217

[58] Field of Search 364/559, 449, 443, 424.01, 424.02, 453, 454, 550, 571.01, 571.02; 318/560, 560, 567; 356/358, 375; 180/187, 189

[56] References Cited

U.S. PATENT DOCUMENTS

4,982,504 1/1981 Soderberg et al. 33/502

5,014,204 5/1981 Kaminura et al. 384/449

5,025,377 8/1981 Kaminura et al. 384/448

5,088,785 11/1981 Kaminura et al. 384/448

5,131,754 7/1992 Hawegawa 358/375

5,323,152 8/1994 Morita 340/998

New Document Note in Memory - Document Annotation

Type: Document Title: New Document Note in Memory

Doc: Ass: Inv: TS: Int: Tit: Inf: Fil: Abs:

Location: Japankia sub-note 2509

Note Window 2508

moving body 1 when it is completed a trip in a certain time

FIG. 25

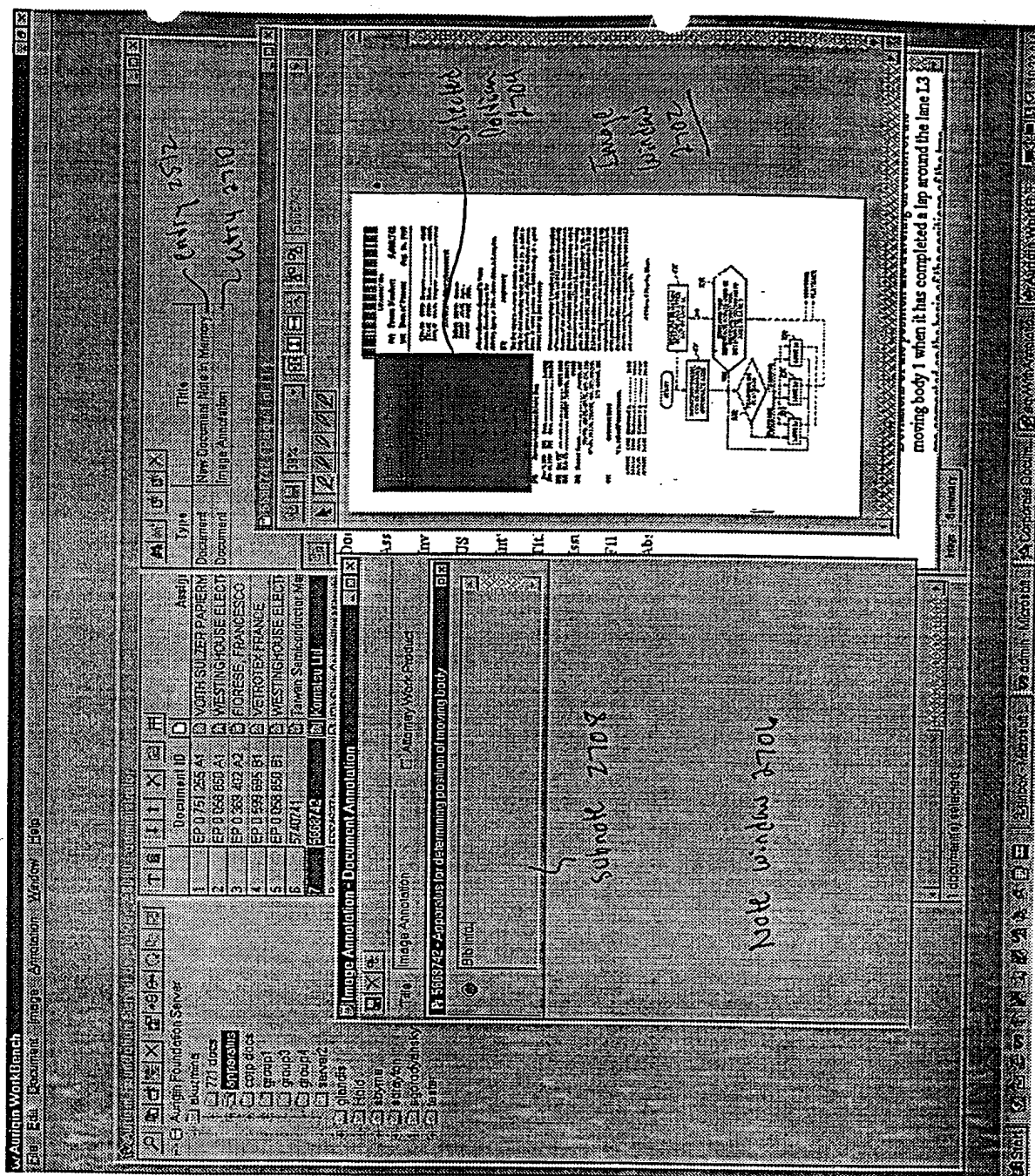


Fig. 27

Contents. Page 24074

Fl. 28

Aurigin Workbench

File Edit Document Image Annotation Window Help

Aurigin Foundation Server ds server 2000 - administrator

Aurigin Foundation Server

71 files
epo-patents
group1
group2
server2
Halo
dyna
traven
spaceability
lars

Document ID | **Type** | **Title** | **Assignee**

Document ID	Type	Title	Assignee
EP 0829 456 A1	pdf modified	N/A	ADONAR COSTRIZIONI MASCHINE AUTOMAT
EP 0816 818 A1			THE PROCTER & GAMBLE COMPANY
EP 0815 817 A1			THE PROCTER & GAMBLE COMPANY
EP 0815 818 A1			THE PROCTER & GAMBLE COMPANY
EP 0815 815 A1			THE PROCTER & GAMBLE COMPANY
EP 0790 457 A1			INGENIEURSBUREAU ORANENWOUD B.V.
EP 0763 571 A1			BASF AKTIENGESellschaft
EP 0751 254 A1			MUTH SULZER PAPERMAN SCHNEIDER OHNER
EP 0661 622 A1			Pfizer Inc.
EP 0653 893 A1			VALMET PAPER MACHINERY INC
EP 0608 911 A1			HEINRICH MACK NACHF
EP 0608 310 A1			HEINRICH MACK NACHF
EP 0416 550 A1			Pfizer Inc.
EP 0413 718 A1			Pfizer Inc.
EP 0410 372 A1			Pfizer Limited
EP 0409 283 A1			Pfizer Inc.
EP 0403 507 A1			Pfizer Inc.
EP 0404 222 A1			Pfizer Inc.
EP 0428 086 A1			Pfizer Inc.
EP 0421 021 A1			Pfizer Limited
EP 0417 568 A1			GEECHAN GROUP PLC
EP 0419 210 A1			Pfizer Inc.
EP 0418 813 A1			Pfizer Inc.
EP 0410 816 A1			Pfizer Limited
EP 0408 015 A1			SMITHKLINE BEECHAM CORPORATION
EP 0397 354 A1			Pfizer Inc.
EP 0397 350 A1			Pfizer Inc.
EP 0393 640 A1			Pfizer Limited
EP 0391 673 A1			Pfizer Limited
EP 0391 825 A1			Pfizer Inc.
EP 0388 054 A1			Pfizer Limited
EP 0379 268 A1			Pfizer Inc.
EP 0378 558 A1			Pfizer Inc.
EP 0378 357 A1			Pfizer Inc.
EP 0365 093 A1			Pfizer Inc.
EP 0364 723 A1			Pfizer Limited
EP 0364 051 A1			Pfizer Inc.
EP 0359 948 A1			Pfizer Limited

document(s) selected

Document ID: pdf modified
Title: pdf2
Author: known
Source: known
Disclosure Date: 01/10/1996
Publication Date: 01/10/1996

data object
page 2400

27/99

Image Window
2902

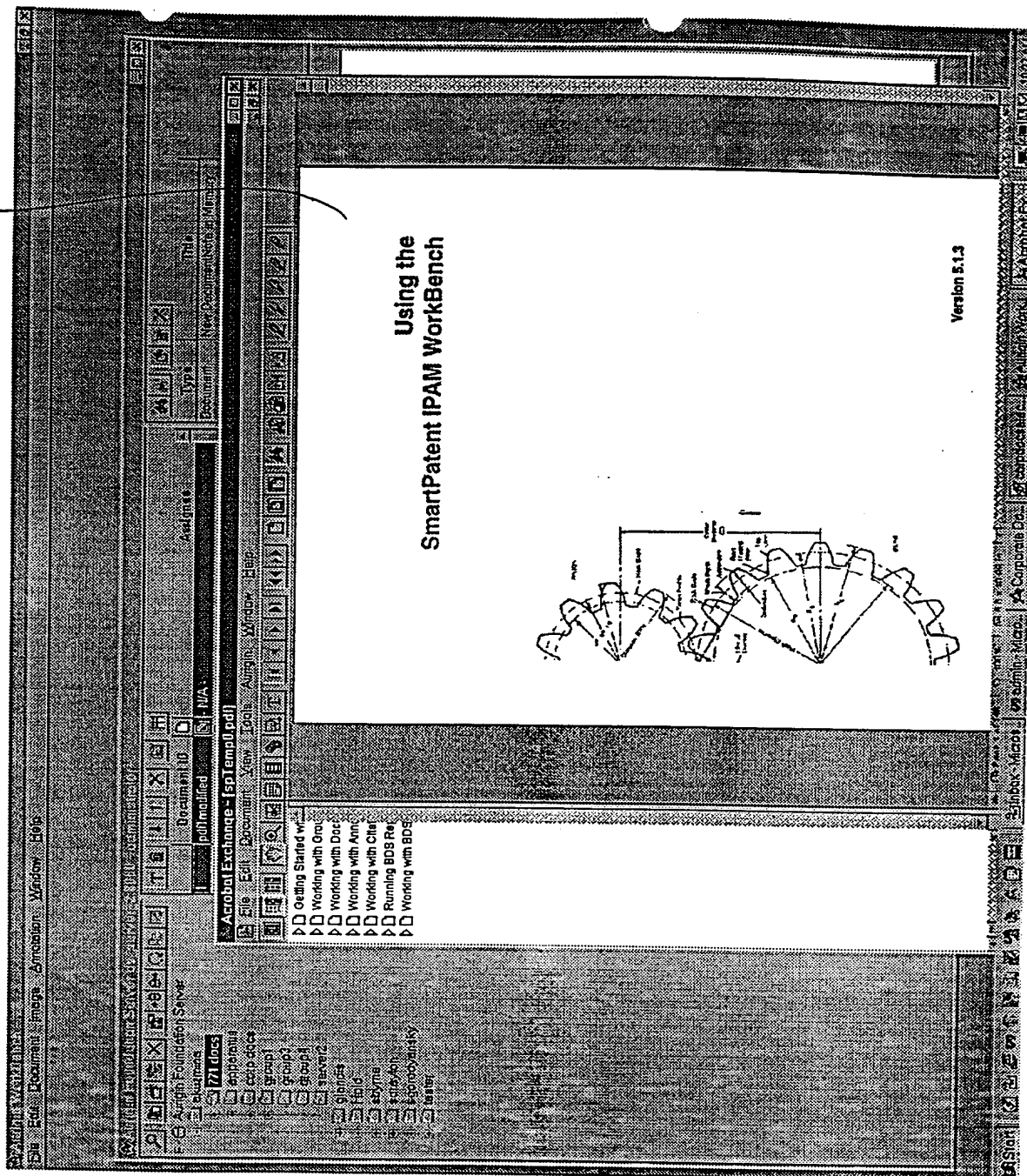
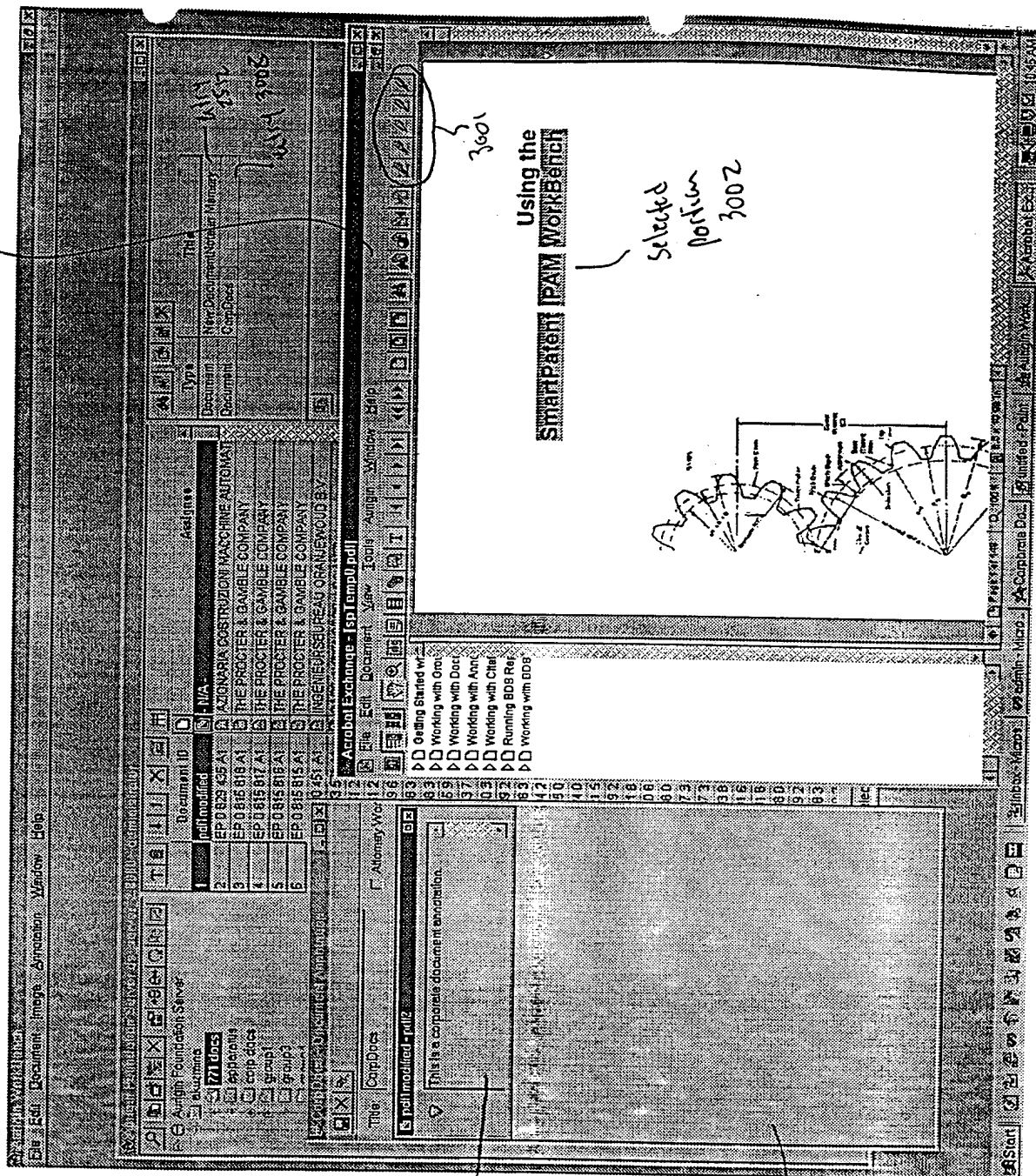


FIG. 29

28/99

Image window 3002



Subtable 3006

FIG. 30

Note window 3004

29/99

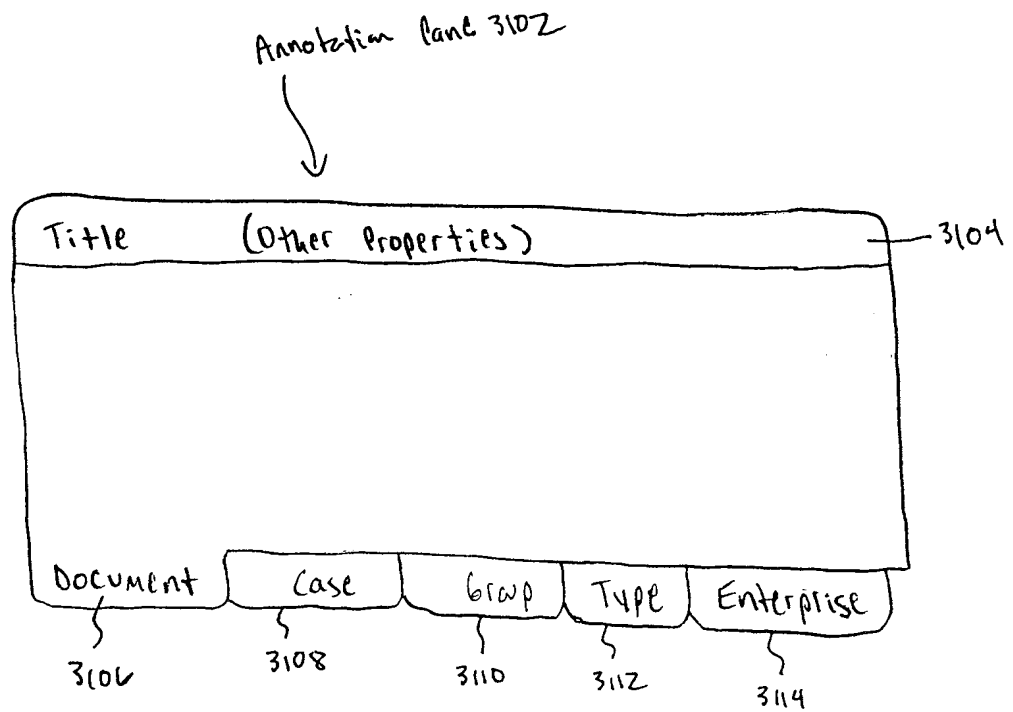


FIG. 31

30/99

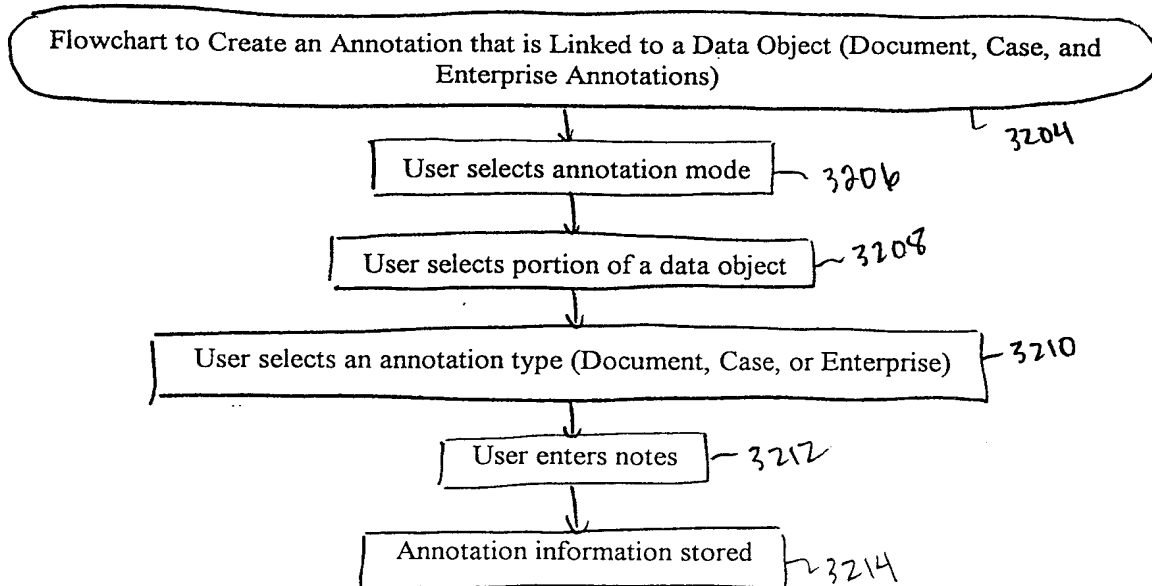
3202
↓

FIG. 32

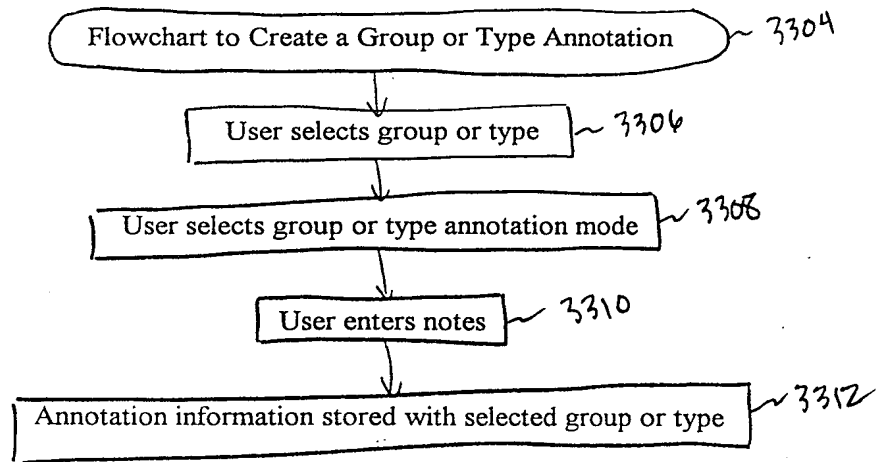
3302
↓

FIG. 33

Search GUI 3402

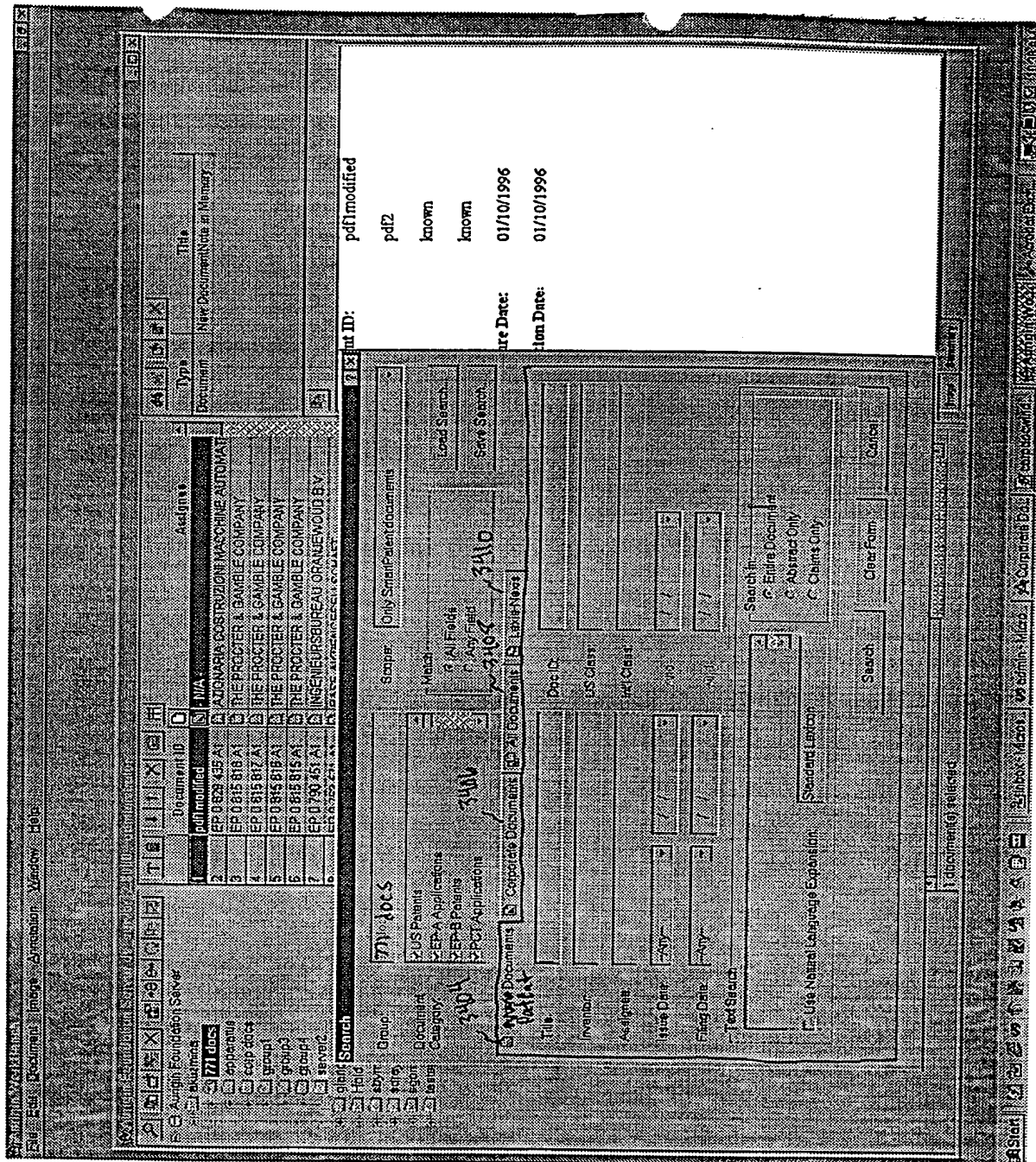
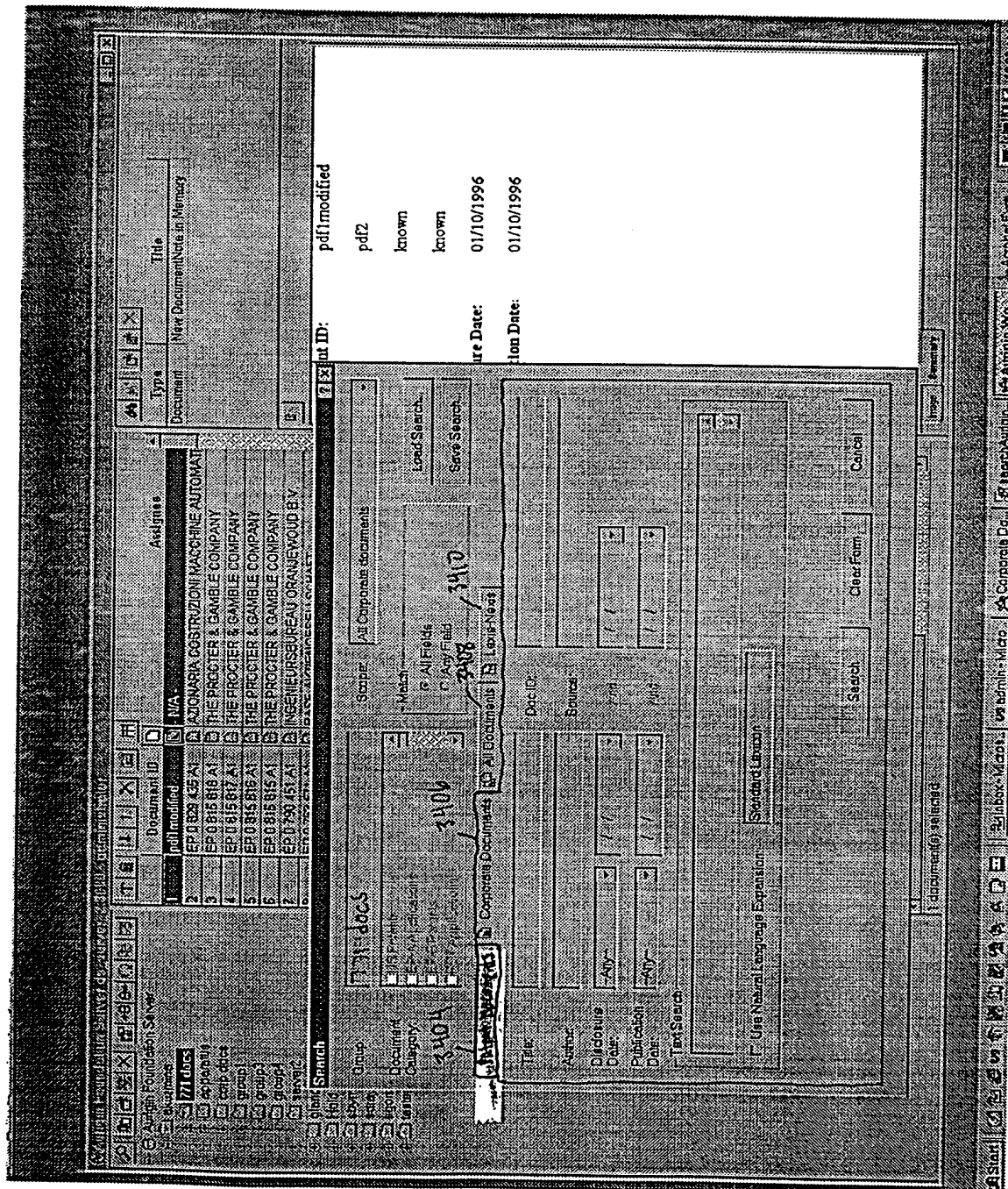


FIG. 34

33/99

FIG. 35



Stack GUI 3402

35/99

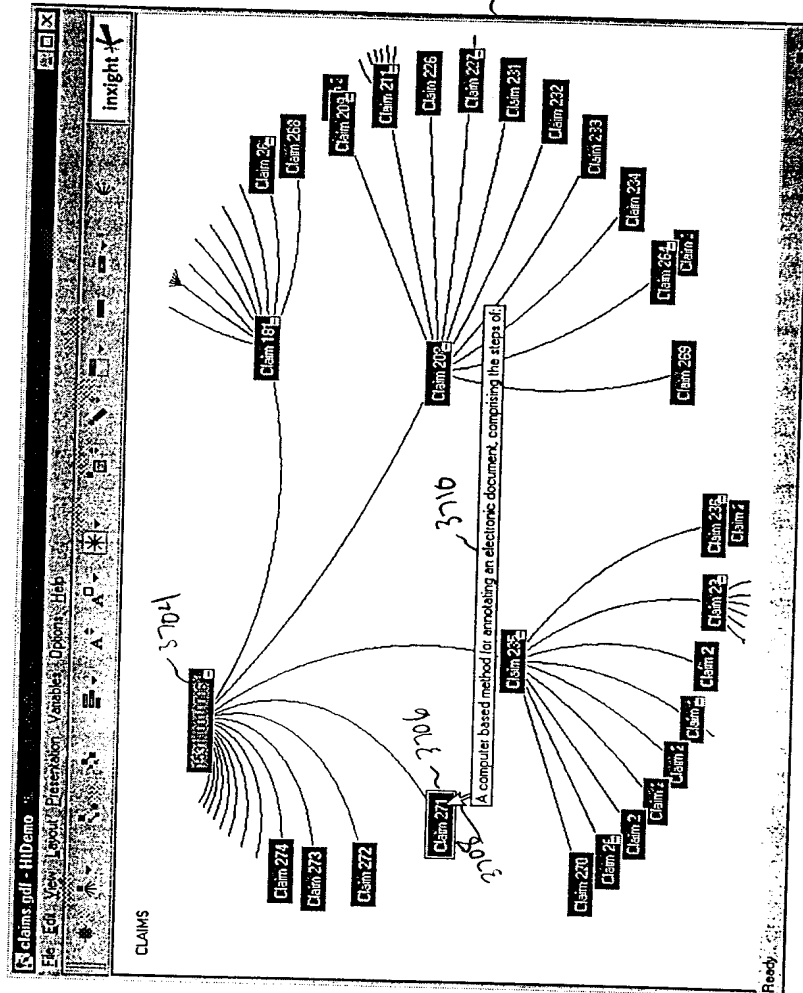


FIG. 37

Claim Tree 3702

Window 3712

36/99

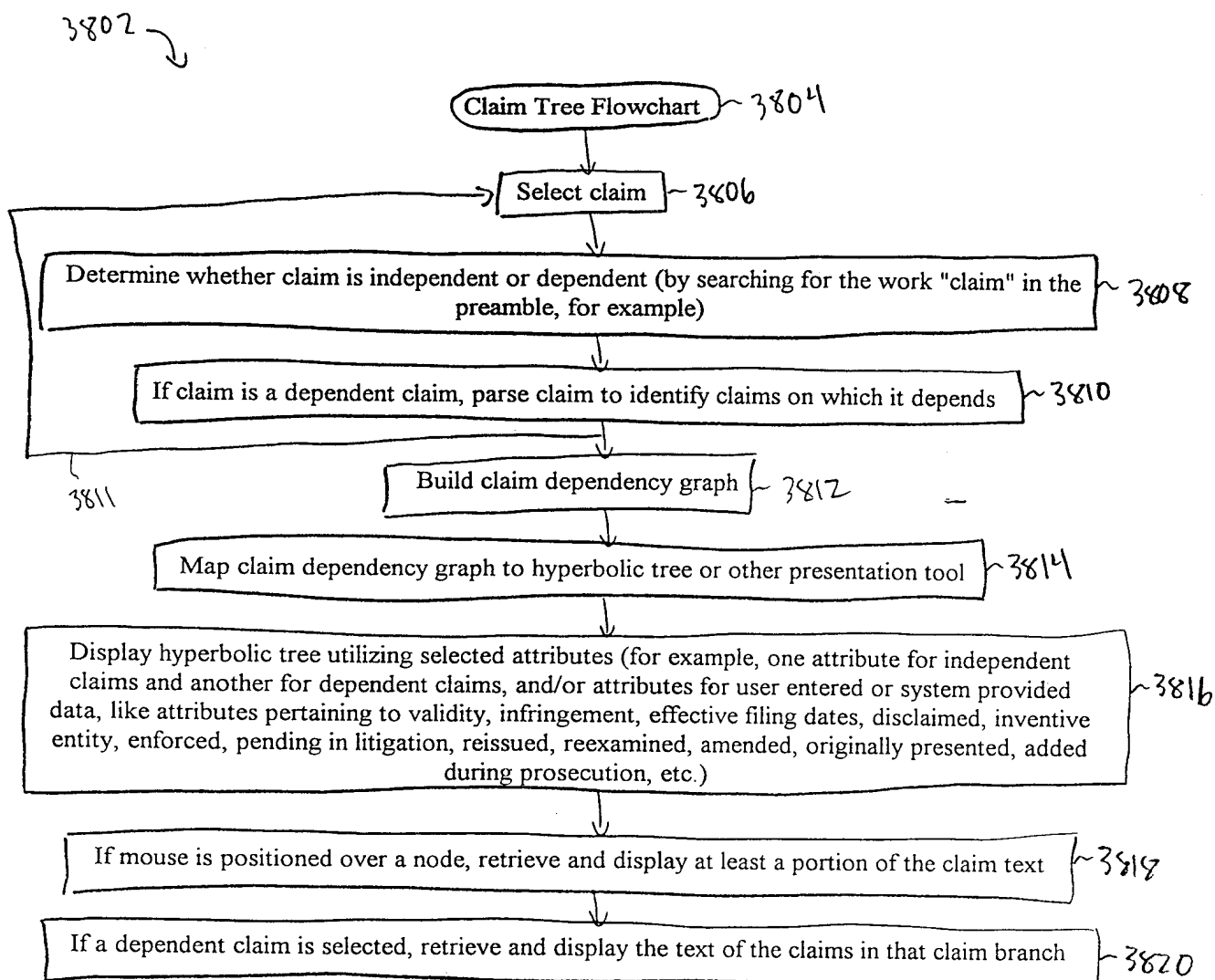


FIG. 38

235. A computer program product, comprising a computer useable medium having computer program logic stored therein, wherein said computer program logic enables a computer to annotate an electronic document, said computer program logic comprising:

document displaying means for enabling the computer to display at least a part of said electronic document;

note displaying means for enabling the computer to display at least one note having at least one note segment linked to at least one portion of said at least a part of said electronic document;

location identification information displaying means for enabling the computer to display in said at least one note, proximate to said at least one note segment, location identification information identifying a location of said at least one portion in said at least a part of said electronic document; and

means for enabling the computer to enable a user to create a new note segment in said at least one note, said new note segment being linked to at least a portion of said at least one portion of said at least a part of said electronic document;

wherein said new note segment and said at least one note segment are nested note segments.

FIG. 39

3902

270. The computer program product of claim 235, wherein said computer program logic further comprises:

means for enabling the computer to enable a user to insert into said at least one note segment at least one of

- (a) a user-selected portion of said electronic document,
- (b) a user-selected portion from another electronic document, and
- (c) user-created work product.

FIG. 40

4602

235+270. A computer program product, comprising a computer useable medium having computer program logic stored therein, wherein said computer program logic enables a computer to annotate an electronic document, said computer program logic comprising:

document displaying means for enabling the computer to display at least a part of said electronic document;

note displaying means for enabling the computer to display at least one note having at least one note segment linked to at least one portion of said at least a part of said electronic document;

location identification information displaying means for enabling the computer to display in said at least one note, proximate to said at least one note segment, location identification information identifying a location of said at least one portion in said at least a part of said electronic document; and

means for enabling the computer to enable a user to create a new note segment in said at least one note, said new note segment being linked to at least a portion of said at least one portion of said at least a part of said electronic document;

wherein said new note segment and said at least one note segment are nested note segments;

wherein said computer program logic further comprises:

means for enabling the computer to enable a user to insert into said at least one note segment at least one of

- (a) a user-selected portion of said electronic document,
- (b) a user-selected portion from another electronic document, and
- (c) user-created work product.

FIG. 41

40/99

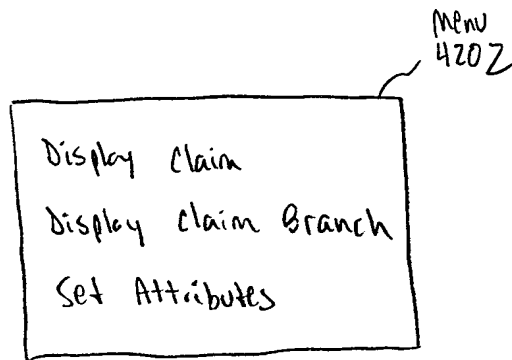


FIG. 4ZA



FIG. 42B

Patent citation Tree 4302
↓

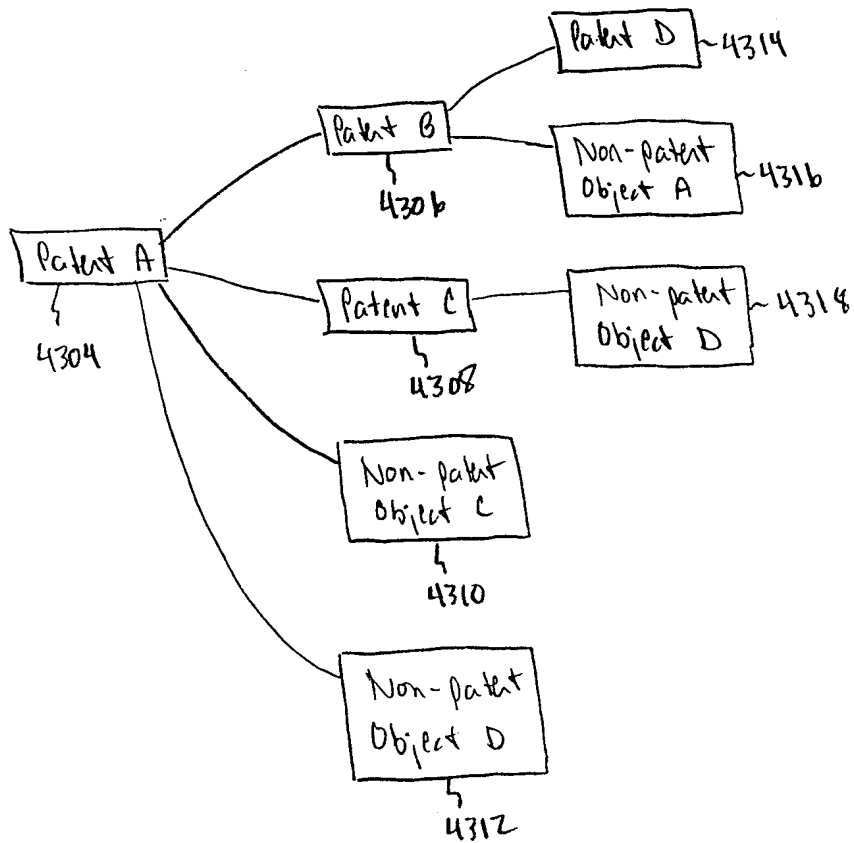


FIG. 43

Patent Ref Table 4402

Ref Patent No	document_id
---------------	-------------

FIG. 44

Non Patent Ref Table 4404

Ref NO	document_id
--------	-------------

FIG. 45

Patent Ref Table 4602

Ref Patent No	document_id
Patent B	Patent A
Patent C	Patent A
Patent D	Patent B

FIG. 46

Non Patent Ref Table 4702

Ref NO	document_id
Non-patent object C	Patent A
Non-patent object D	Patent A
Non-patent object A	Patent B
Non-patent object D	Patent C

FIG. 47

Data Object Family Table 4802 ↴

From 4804	To 4806	Relationship Type 4808

FIG. 48

Data Object Family Table 4902 ↓

From	To	Relationship Type	
Patent A	Patent B	Backward citation	~ 4904
Patent A	Patent C	Backward citation	~ 4906
Patent A	Non-Patent Object C	Backward citation	~ 4908
Patent A	Non-Patent Object D	Backward citation	~ 4910
Patent B	Patent D	Backward citation	~ 4912
Patent B	Non-Patent Object A	Backward citation	~ 4914
Patent C	Non-Patent Object D	Backward citation	~ 4916
Patent B	Patent A	Forward citation	~ 4918
Patent D	Patent B	Forward citation	~ 4920
Non-Patent Object A	Patent B	Forward citation	~ 4922
Patent C	Patent A	Forward citation	~ 4924
Non-Patent Object D	Patent C	Forward citation	~ 4926
Non-Patent Object C	Patent A	Forward citation	~ 4928
Non-Patent Object D	Patent A	Forward citation	~ 4930

FIG. 49

Relationship Type Table 5002 ↴

Rel_Type_ID	Relationship Type Name	Grouping
Rel_Type_001	Backward Citation	
Rel_Type_002	Forward Citation	
Rel_Type_003	Original (Parent)	G1
Rel_Type_004	Continuation	G1
Rel_Type_005	Continuation-In-Part	G1
Rel_Type_006	Divisional	G1
Rel_Type_007	Reissue	G1
Rel_Type_008	Reexamination	G1
Rel_Type_009	Design	G1
Rel_Type_010	Utility	G1
Rel_Type_011	SIR (Statutory Invention Registration)	G1
Rel_Type_012	Defensive Publication	G1
Rel_Type_013	Invention Disclosure Materials	G1
Rel_Type_014	Priority document	G1, G2
Rel_Type_015	PCT	G1, G2
Rel_Type_016	Foreign National Application - Canada	G1, G2
Rel_Type_017	Foreign National Application - Australia	G1, G2
Rel_Type_018	Foreign National Application - United Kingdom	G1, G2
Rel_Type_019	Foreign National Application - Taiwan	G1, G2

FIG. 50A

Rel_Type_020	Foreign National Application - Japan	G1, G2
Rel_Type_021	Foreign National Application - Mexico	G1, G2
Rel_Type_022	Foreign National Application - Brazil	G1, G2
Rel_Type_023	A1	G1, G2
Rel_Type_024	A2	G1, G2
Rel_Type_025	A3	G1, G2
Rel_Type_026	A4	G1, G2
Rel_Type_027	Technically related - Base	G3
Rel_Type_028	Technically related - Improvement	G3
Rel_Type_029	Claim - Dominant	
Rel_Type_030	Claim - Subservient	
Rel_Type_031	Claim - Improvement	
Rel_Type_032	Claim - Independent	
Rel_Type_033	Claim - Dependent	
Rel_Type_034	Trademark	G1
Rel_Type_035	Copyright	G1
Rel_Type_036	Trade Secret	G1

FIG. 50B

Patent Family Chronology 5102

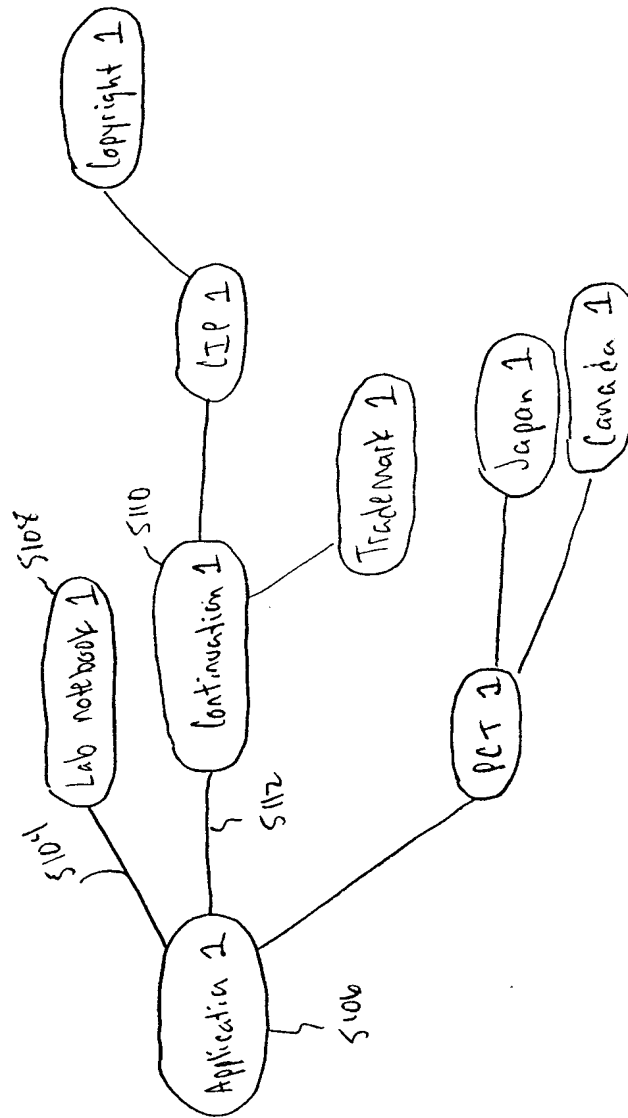


Fig. 51

Data Object Family Table 5202 ↓

From	To	Relationship Type	
Application 1	Lab Notebook 1	Invention Disclosure Materials	5204
Application 1	Continuation 1	Continuation	5206
Continuation 1	CIP 1	Continuation-In-Part	5208
CIP 1	Copyright 1	Copyright	5210
Continuation 1	Trademark 1	Trademark	5212
Application 1	PCT 1	PCT	5214
PCT 1	Japan 1	Foreign National Application - Japan	5216
PCT 1	Canada 1	Foreign National Application - Canada	5218

FIG. 52

Assignee
Technology
Patent Family 5302 ↘

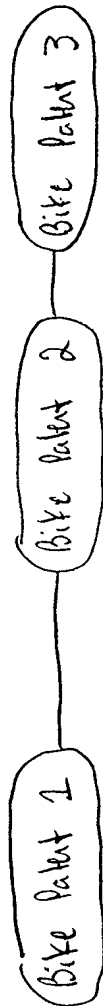


FIG. 53

Data Object Family Table 5402 ↓

From	To	Relationship Type	
Bike Patent 1	Bike Patent 2	Technically related - Improvement	5404
Bike Patent 2	Bike Patent 1	Technically related - Base	5406
Bike Patent 2	Bike Patent 3	Technically related - Improvement	5408
Bike Patent 1	Bike Patent 2	Claim - Subservient	5410
Bike Patent 1	Bike Patent 2	Claim - Improvement	5412
Bike Patent 2	Bike Patent 1	Claim - Dominant	5414

FIG. 54

5502 ↘

Flowchart for Generating Relationship Tables

5504

Obtain relationship information. In some cases, relationship information is contained in existing tables, such as the PatentRef table (FIG. 44) and the NonPatent Ref Table (FIG. 45). In other cases, relationship information can be obtained by parsing/analyzing information in the corresponding data objects, such as by analyzing the bibliographic information of documents. In other cases, relationship information is obtained from client databases (such as a company's patent docket, bill-of-material information, etc.). In other cases, relationship information can be obtained by key word searching of pertinent data objects, such as searching for the term "microprocessor" to identify patents and patent applications and other data objects that relate to microprocessors. More generally, relationship information can be obtained from (1) referential integrity; (2) relational tables; (3) searching the database(s); (4) text searches; and/or (5) manual analysis (see FIG. 56).

Populate relationship table(s) using relationship information.

5508

5506

FIG. 55

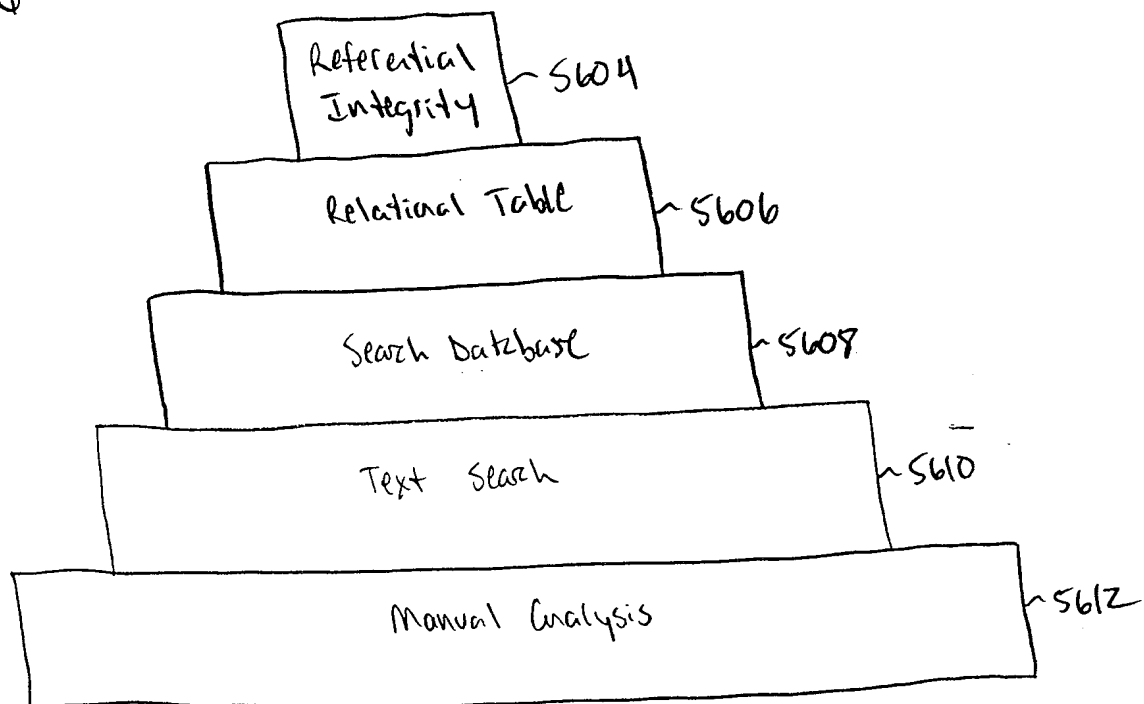
5602
↓

FIG. 56

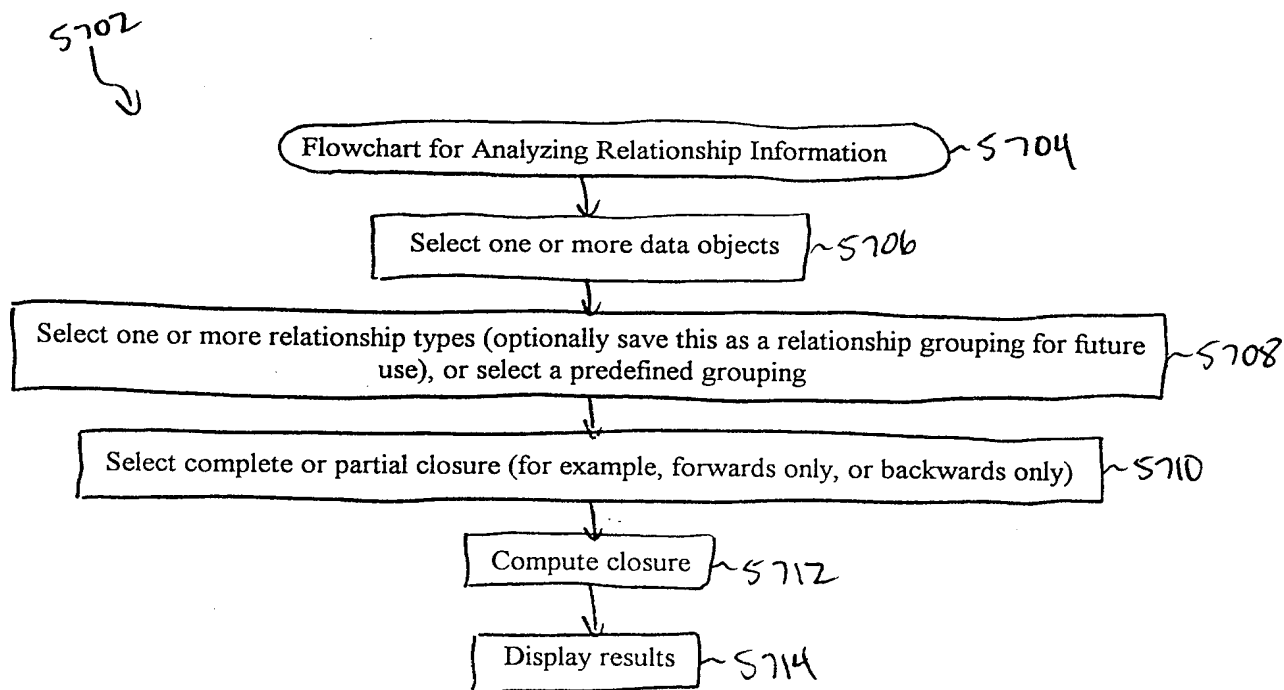


FIG. 57A

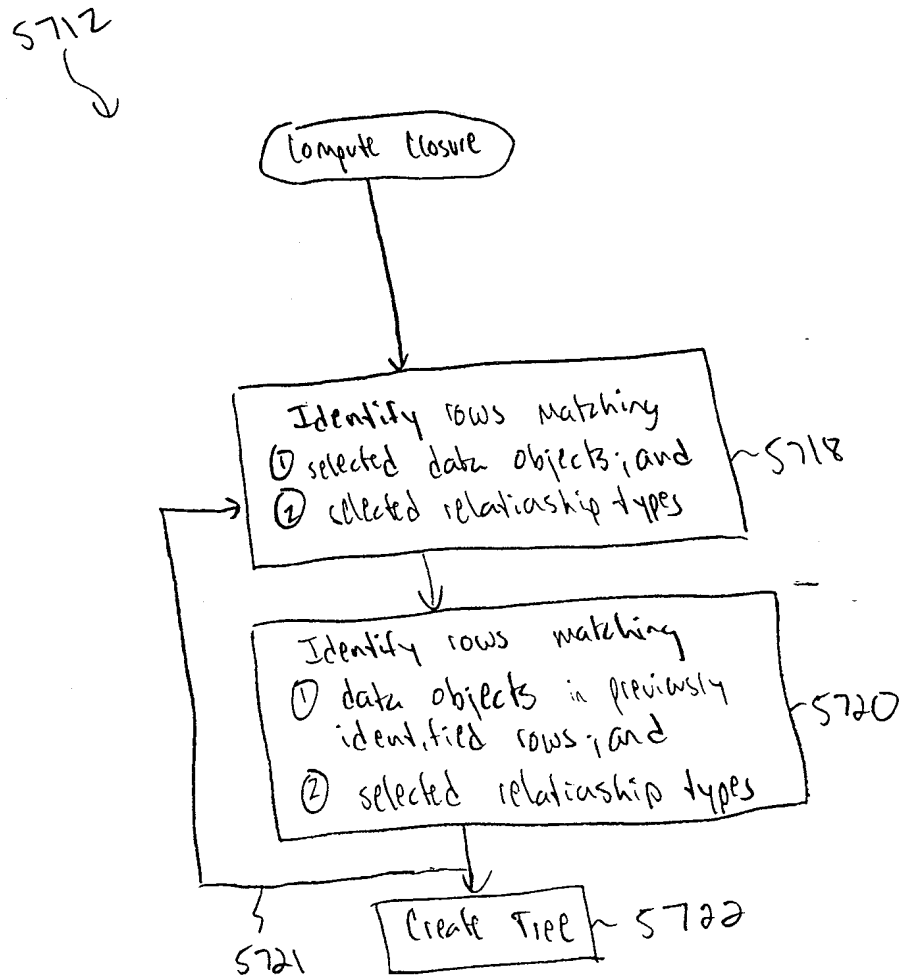


FIG. 57B

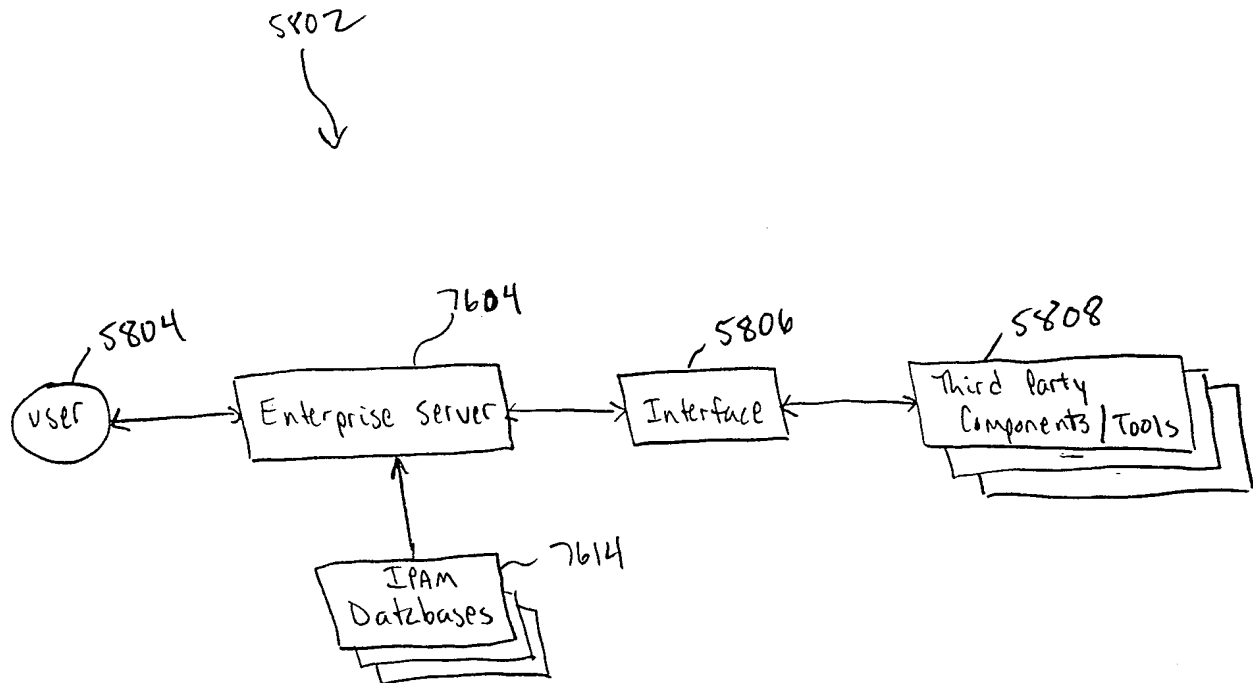


FIG. 58

57/99

5902 ↘

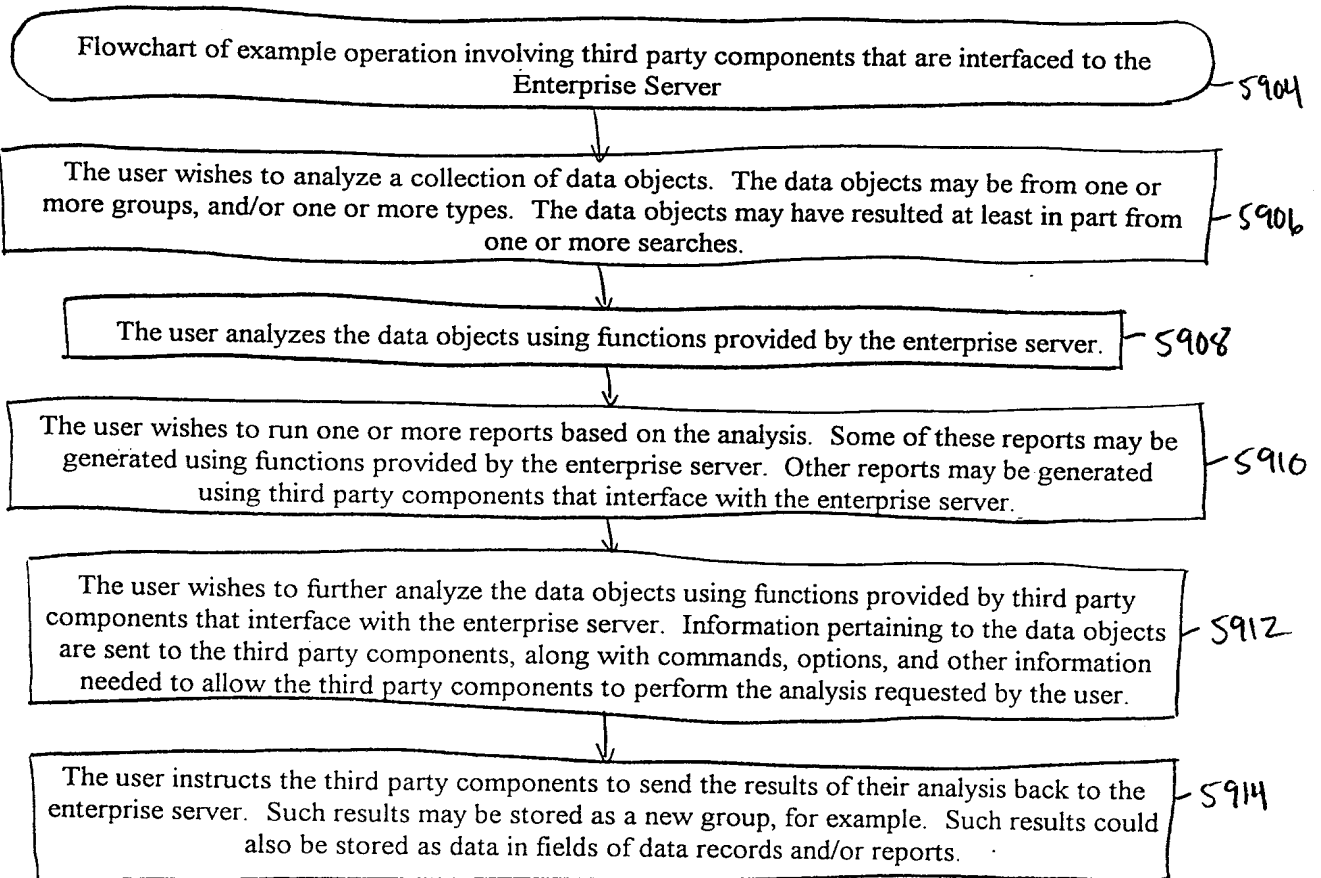


FIG. 59

58/99

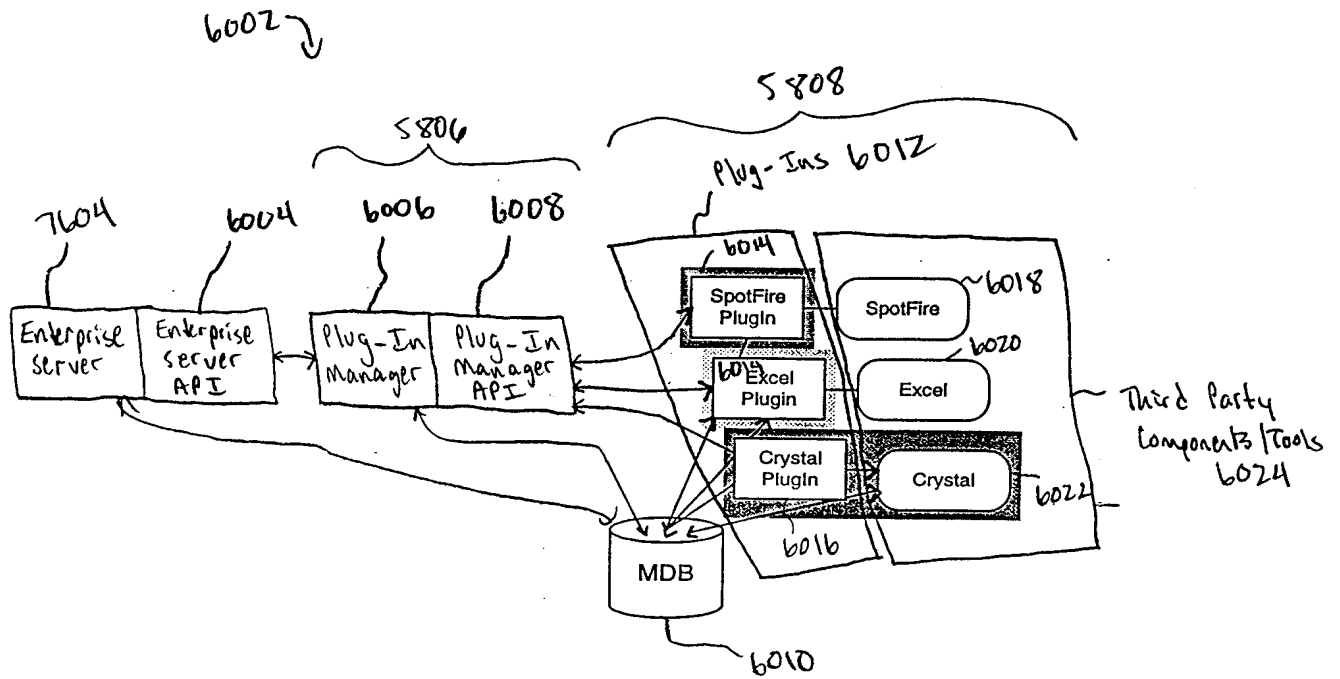


FIG. 60

59/99

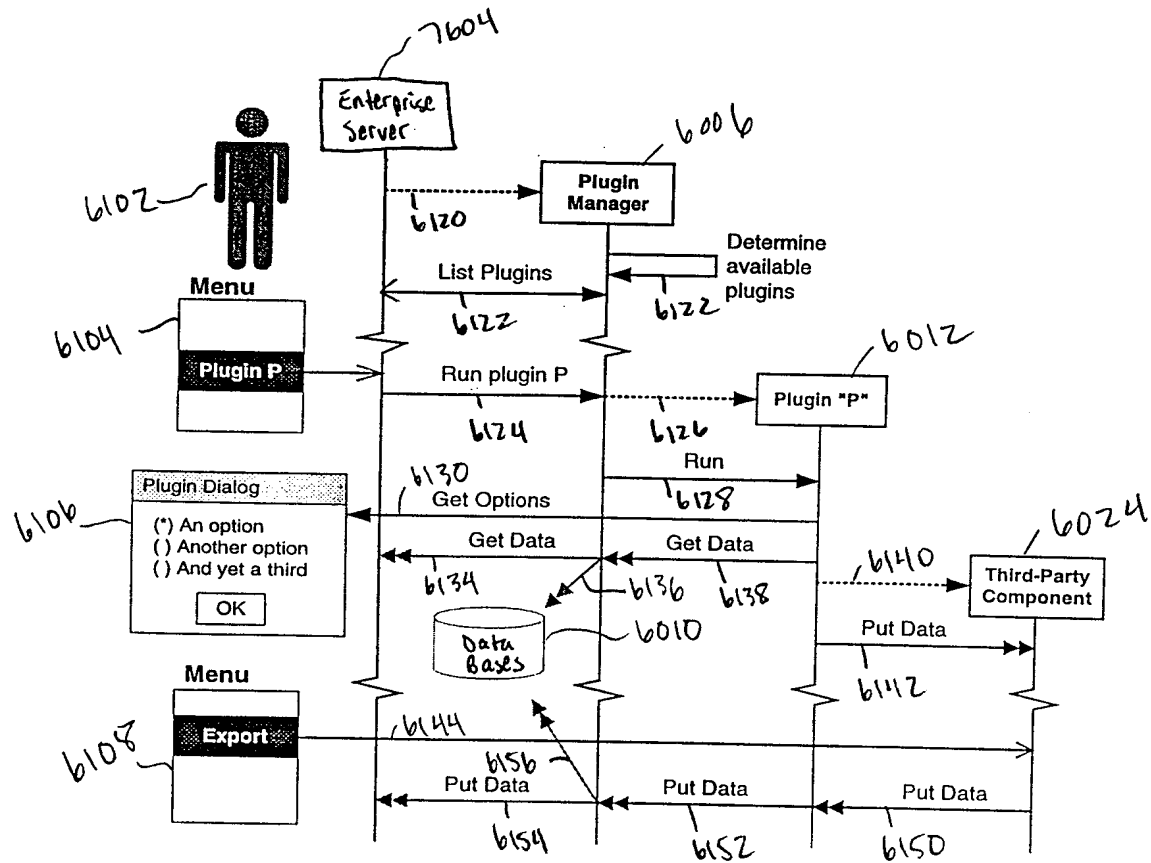


FIG. 61

60/99

6202

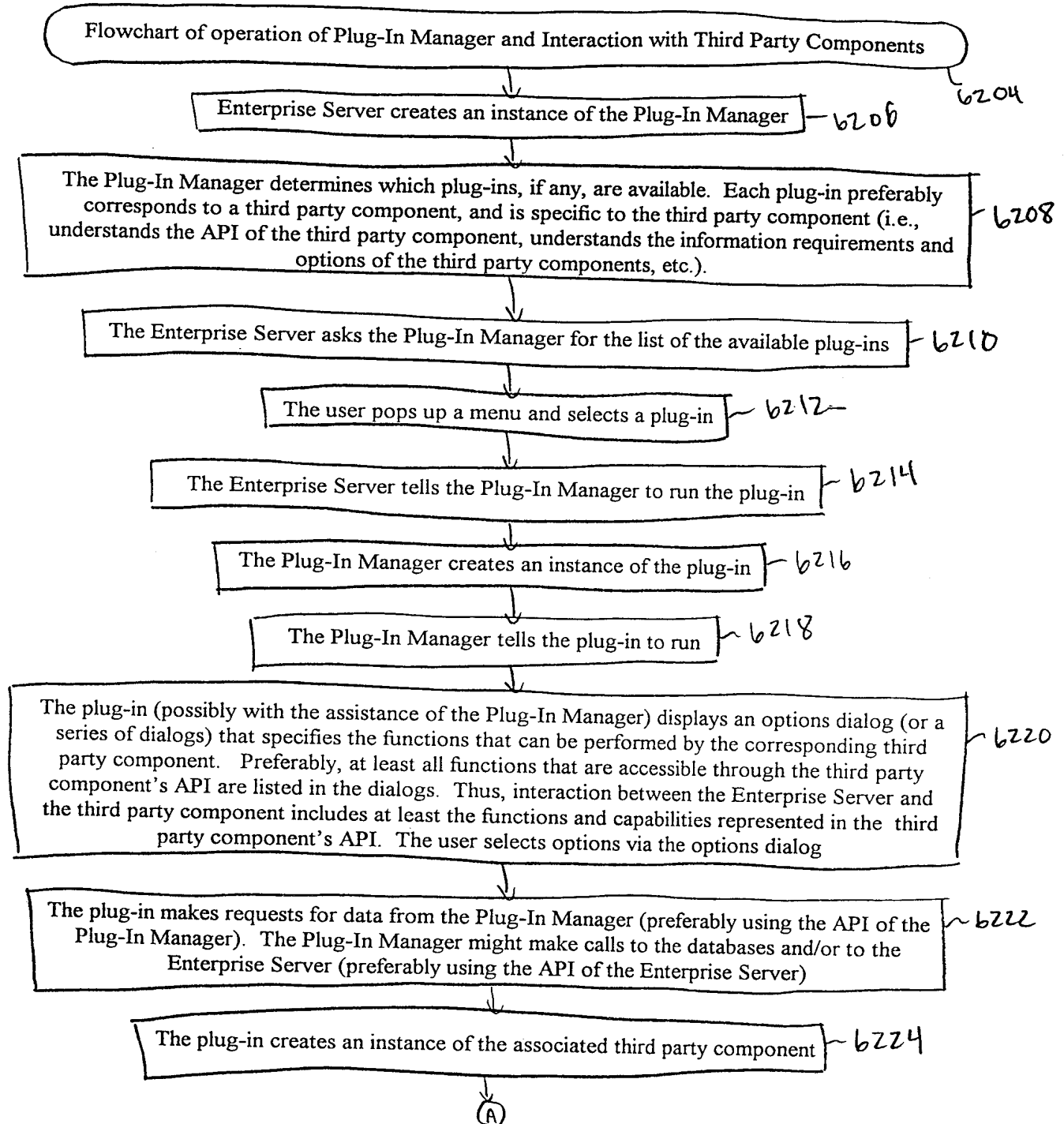


FIG. 62A

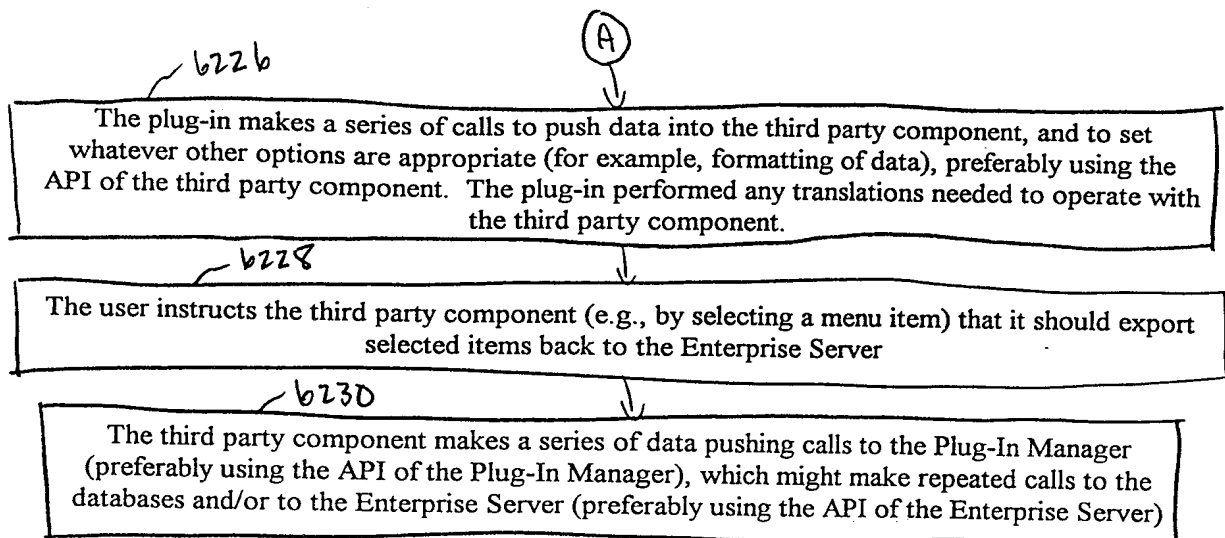


FIG. 62B

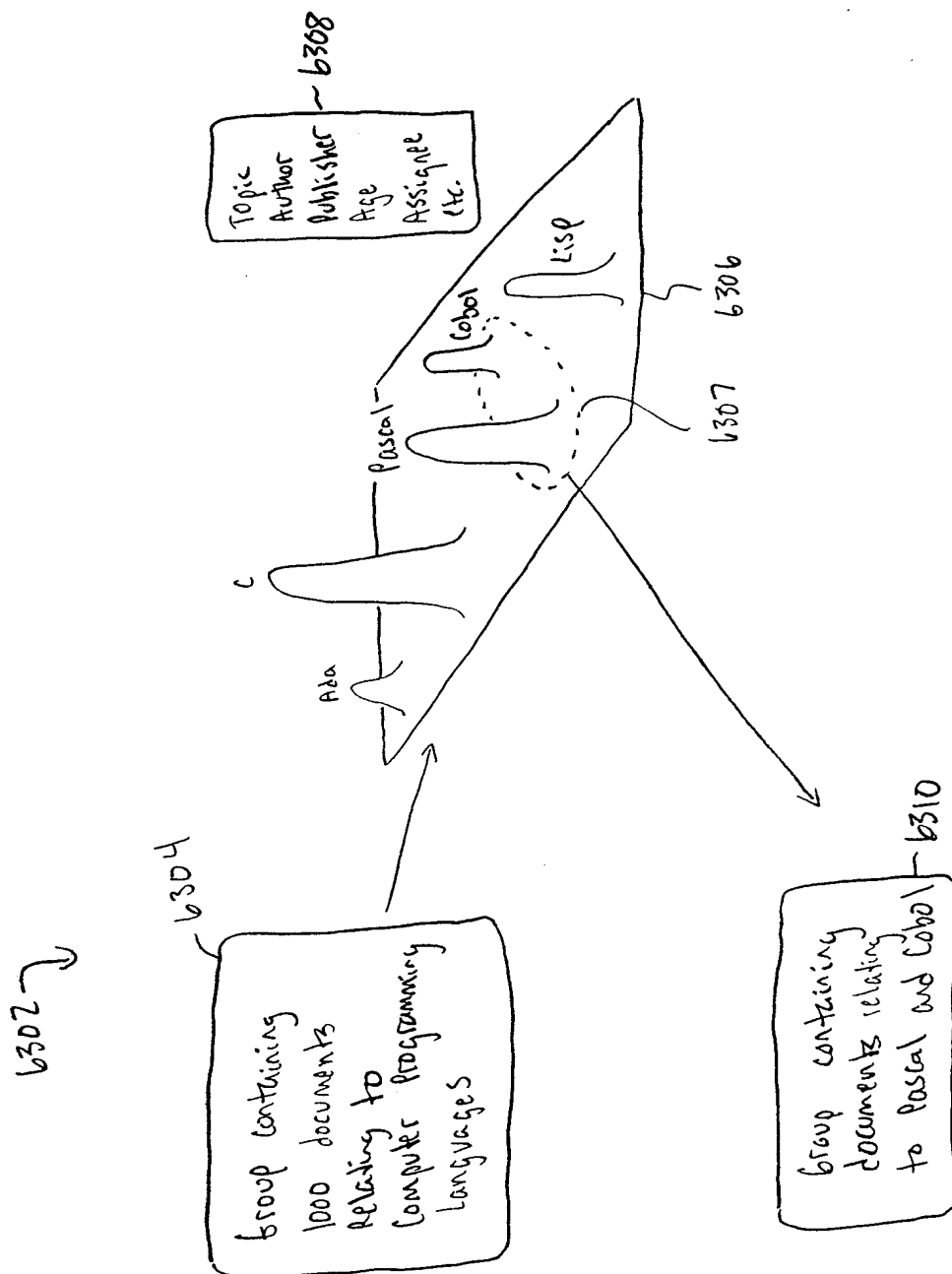


FIG. 63

b402 ↘

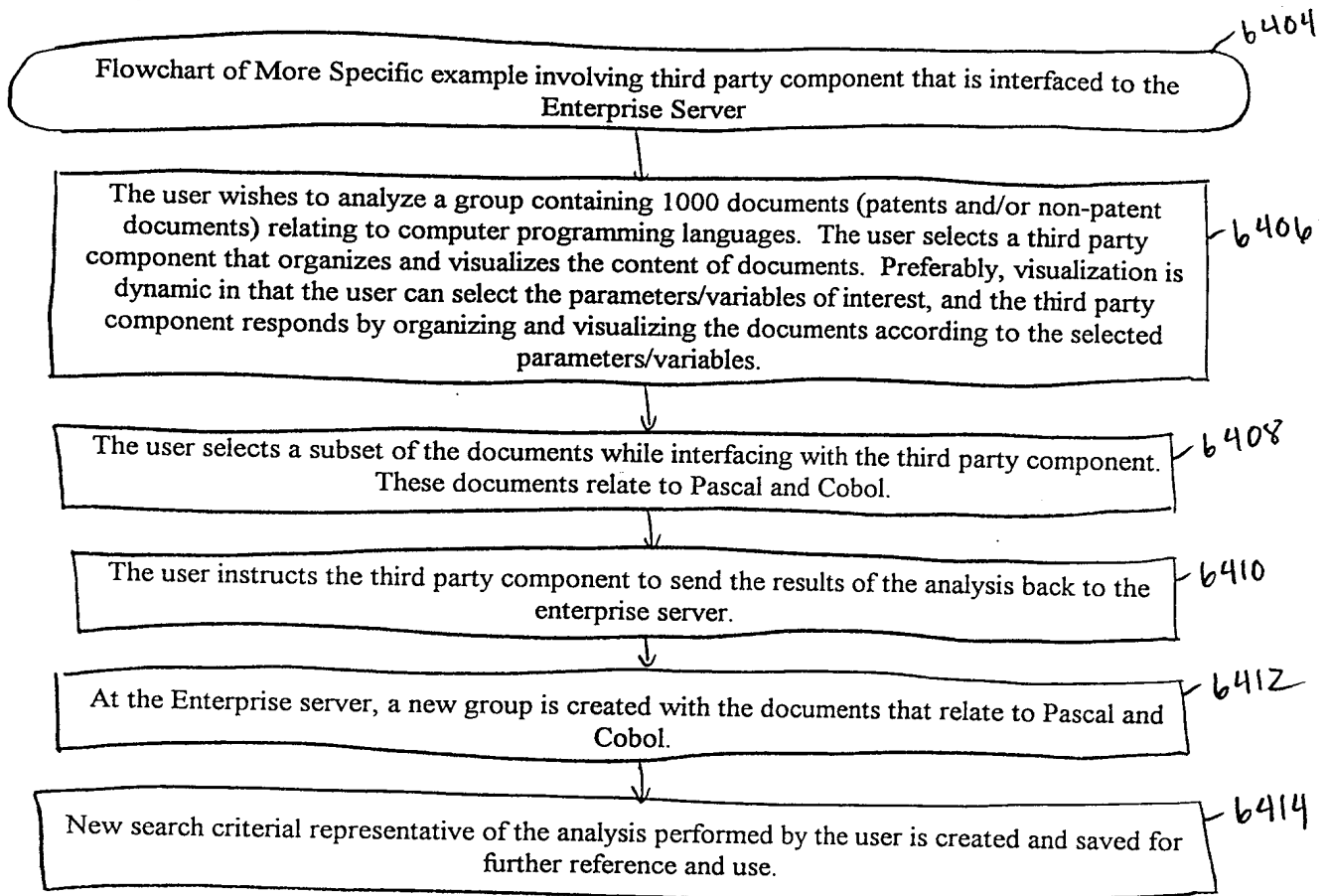


FIG. 64

64/99

6502

6508

6506

6510

6512

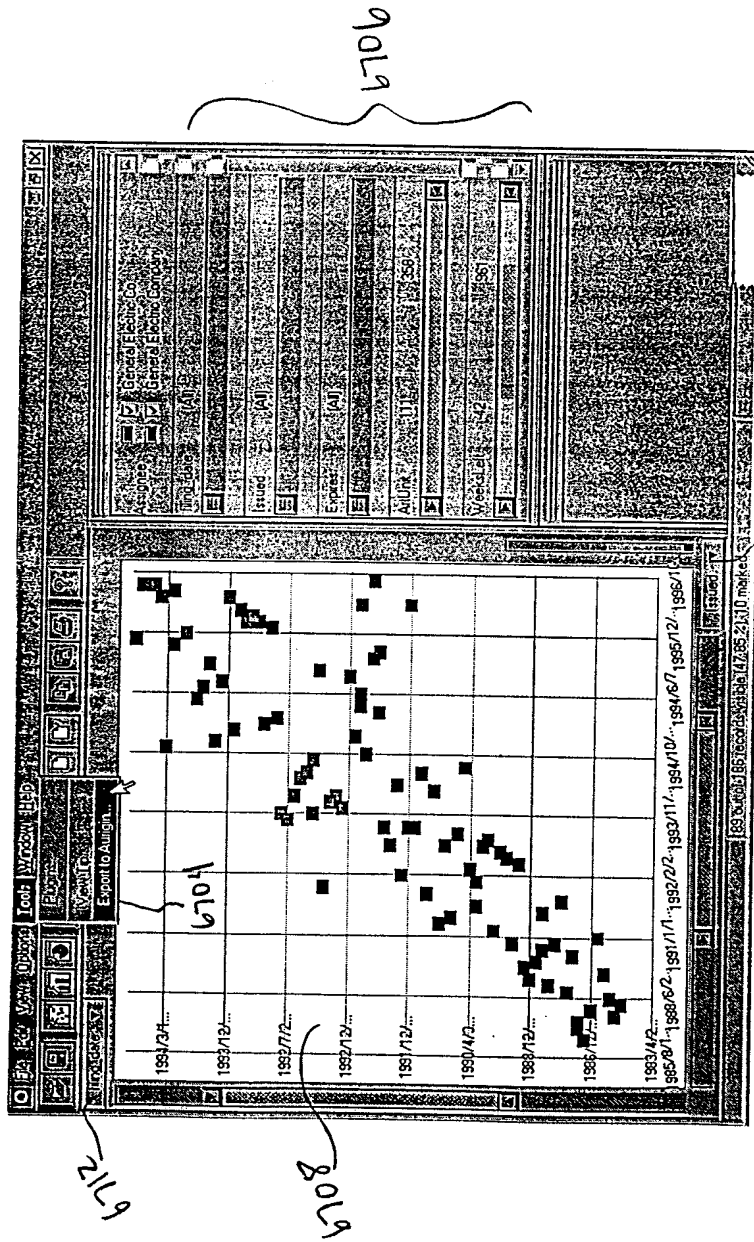
What is claimed is: 1. A system for [114]

Claim-Subject Matter	
All	Assignee Pending Law Firm Examiner Art Unit
	Assignee Pending Law Firm Examiner Art Unit

6505

66/99

6702



6710

FIG. 6

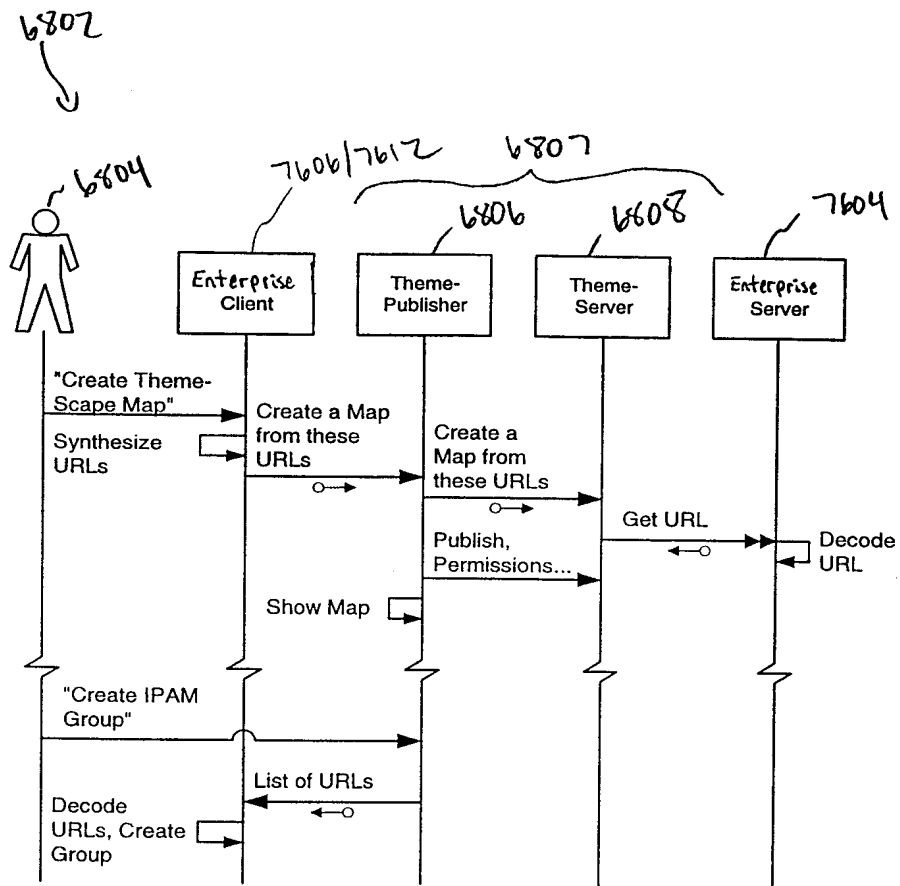


FIG. 68

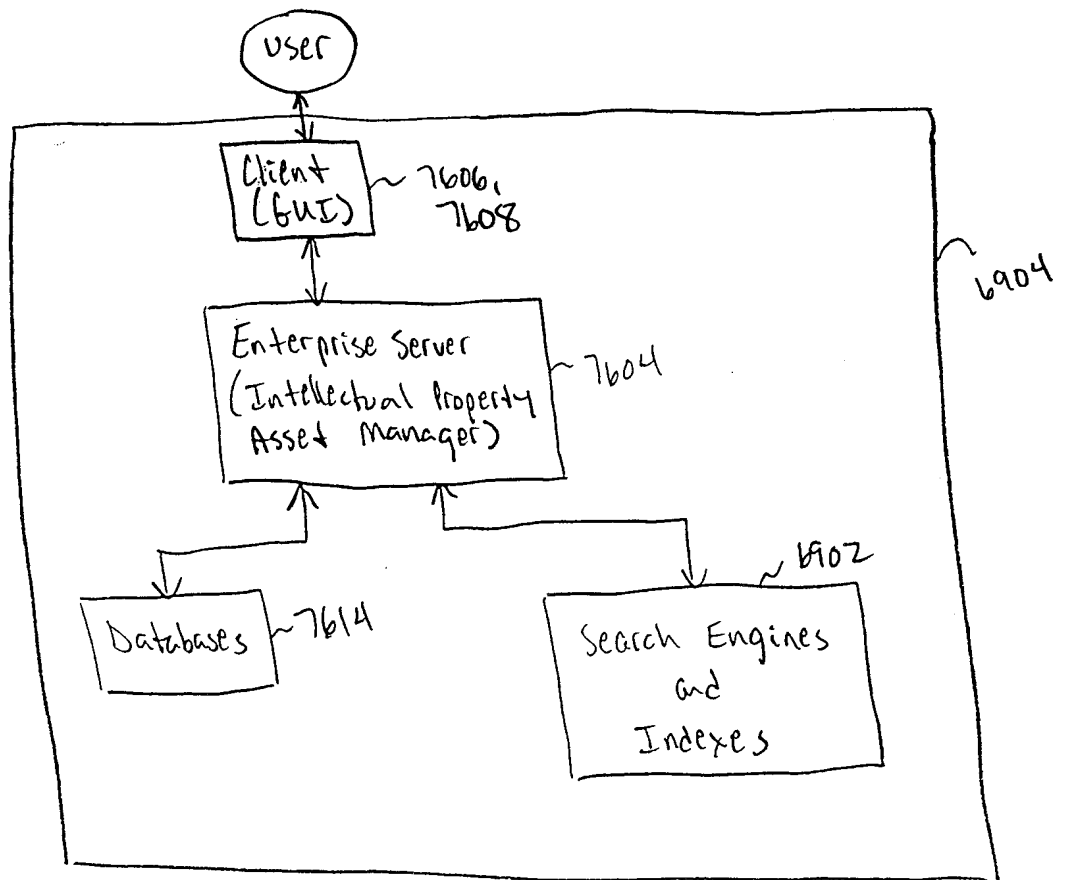


FIG. 69

69/99

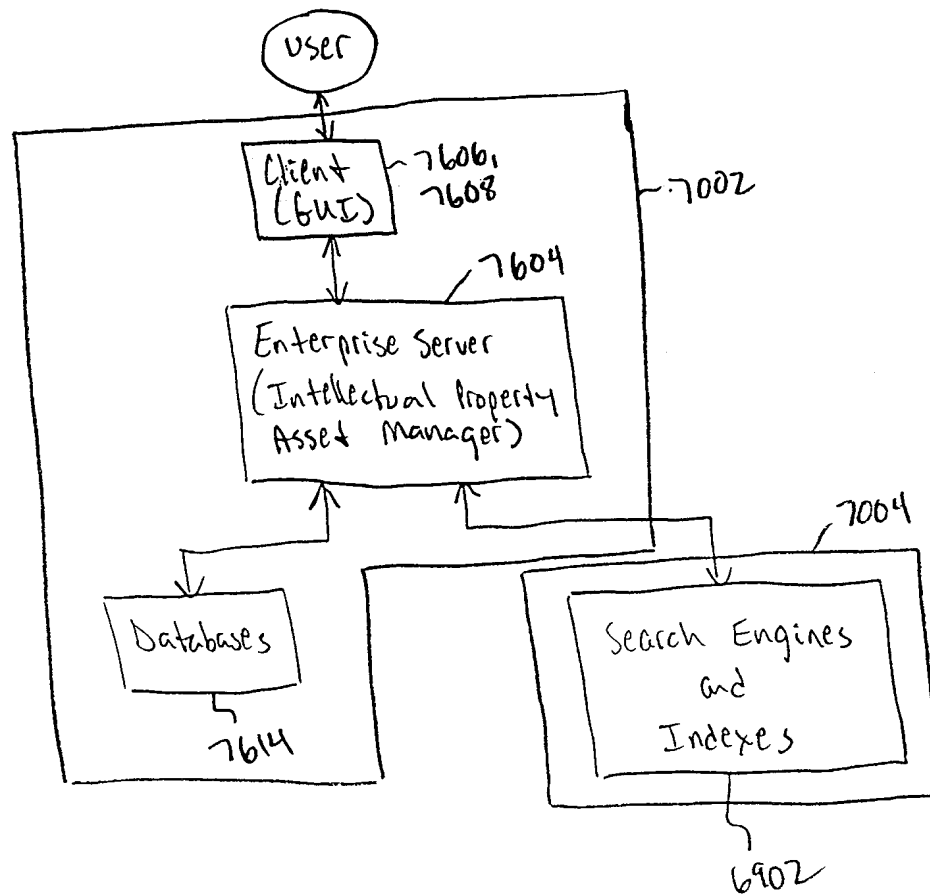


FIG. 70

70/99

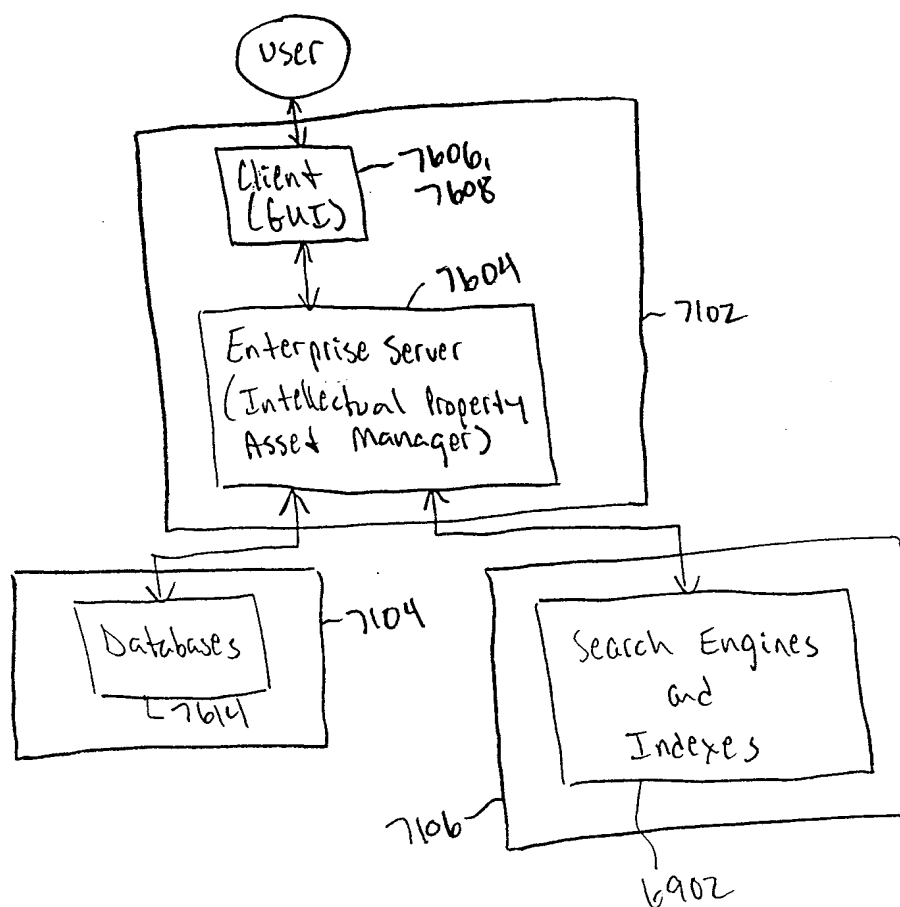


FIG. 71

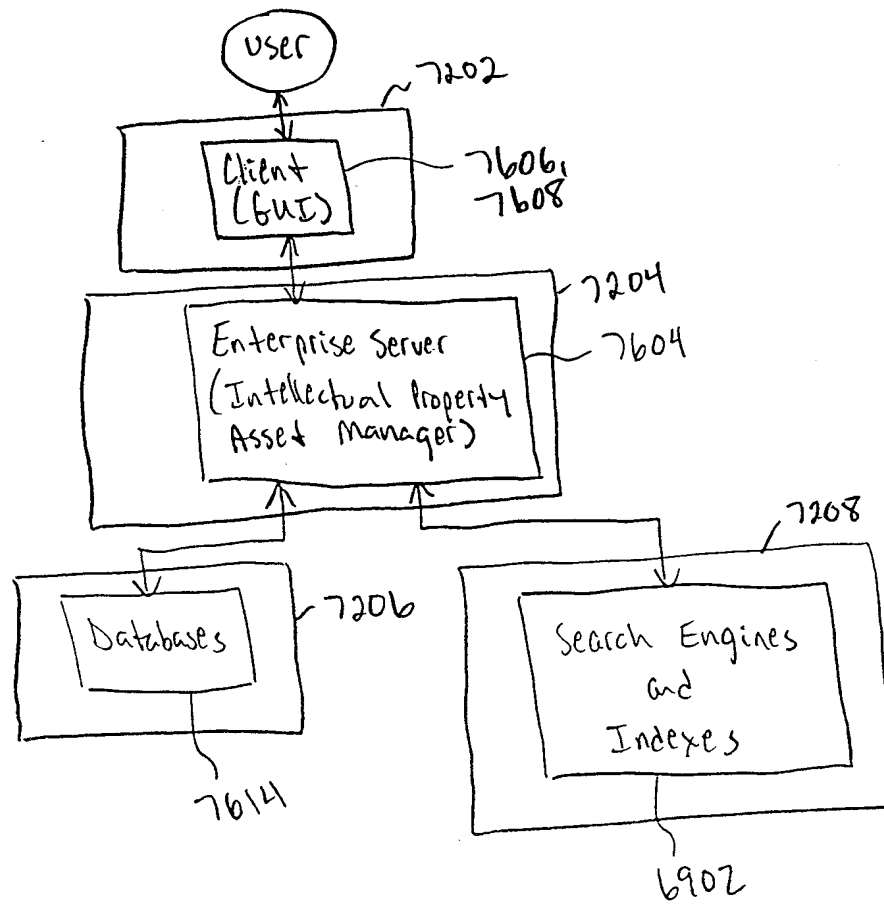


FIG. 72

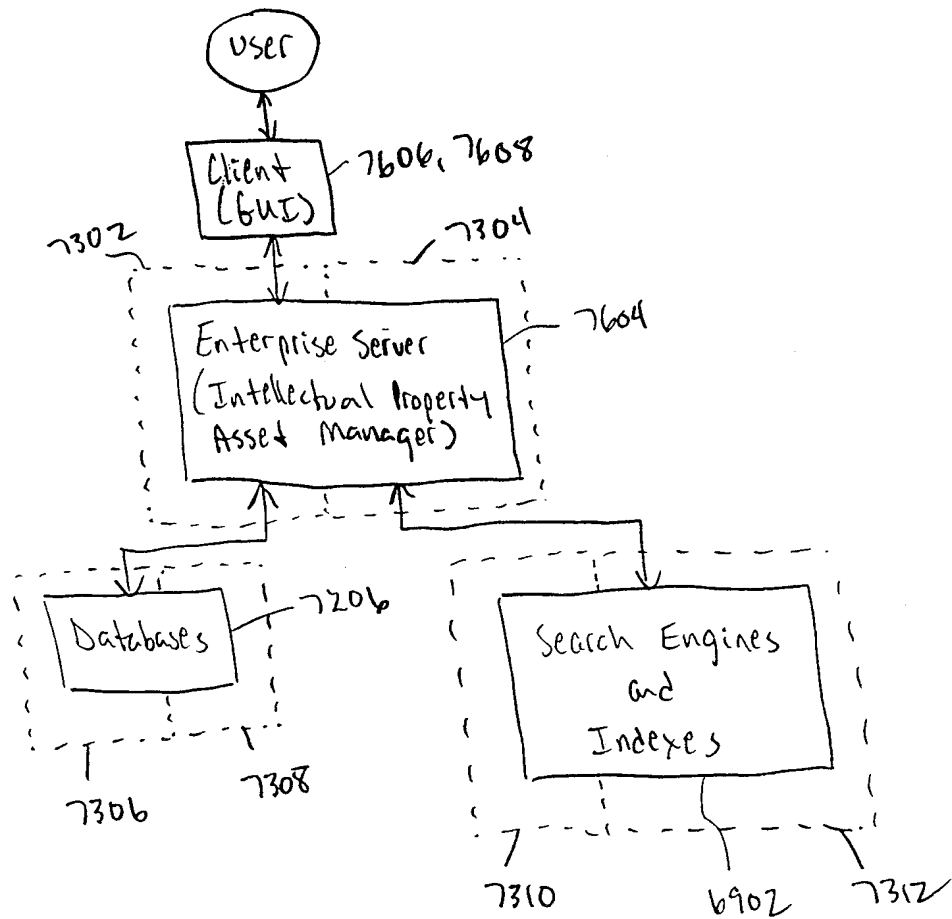


FIG. 73

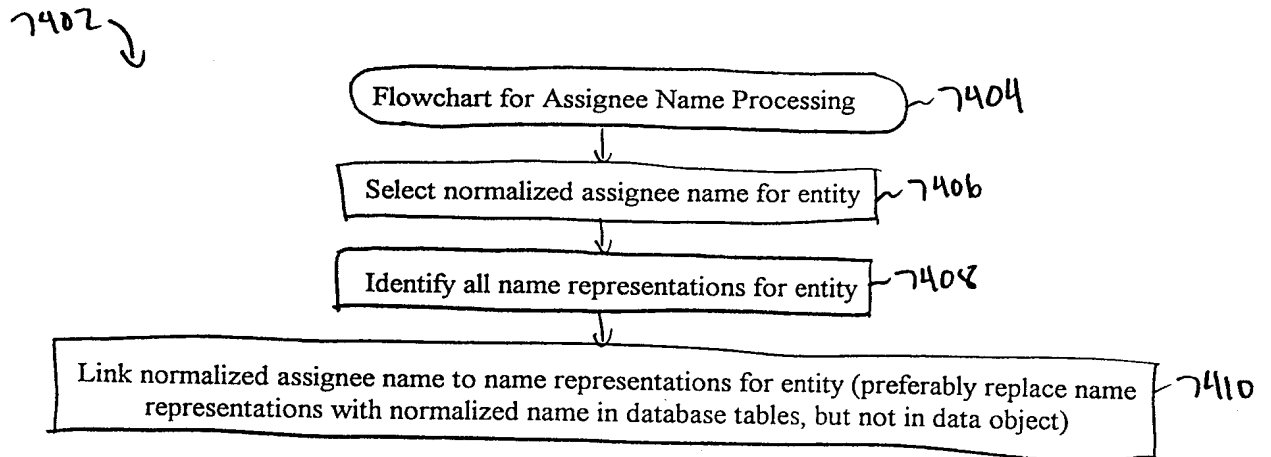


FIG. 74

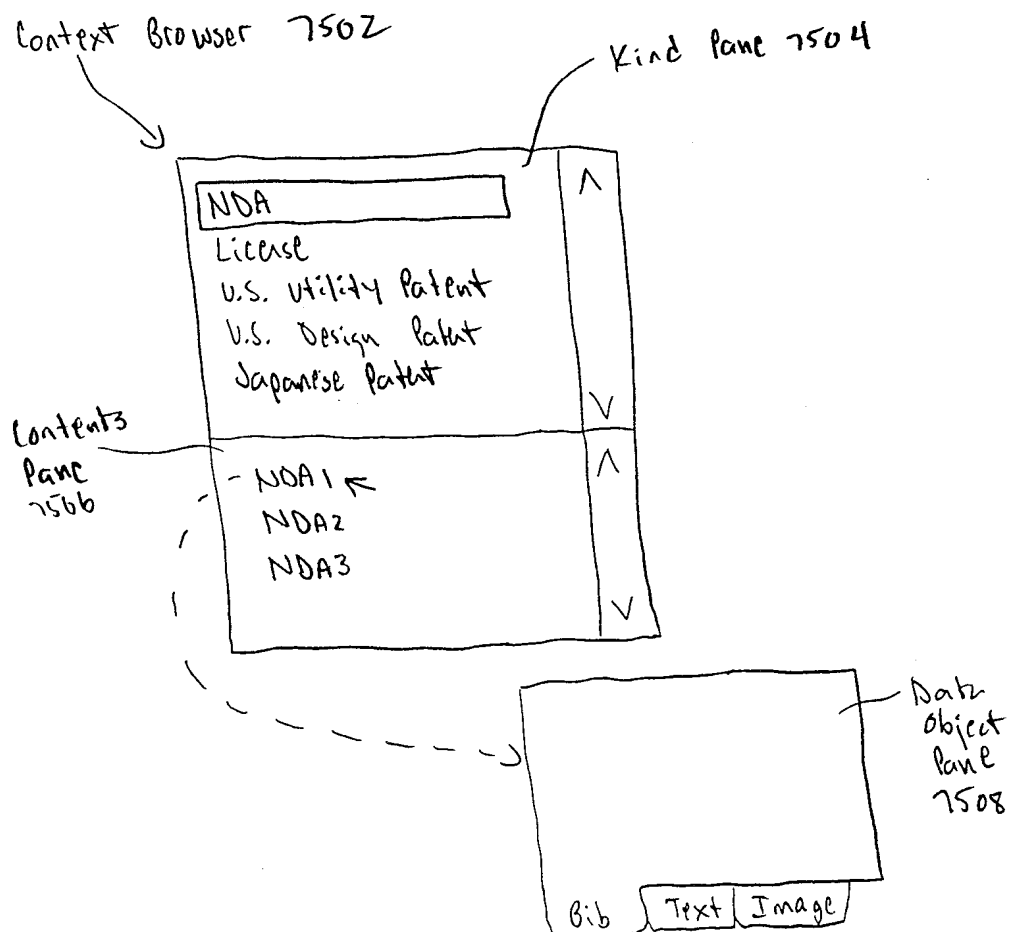


FIG. 75

75/99

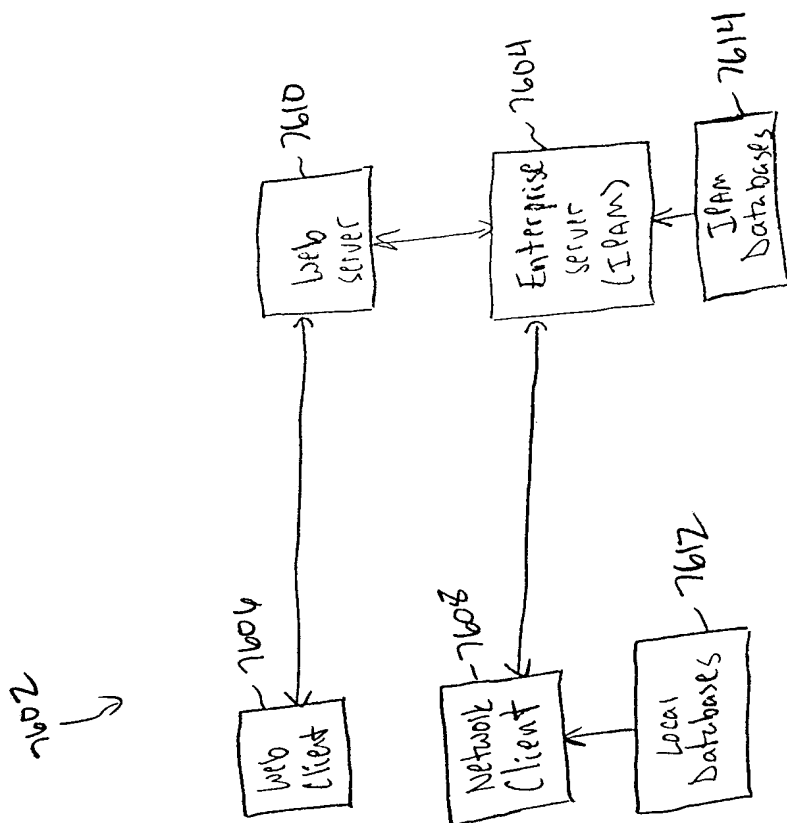


FIG. 7b

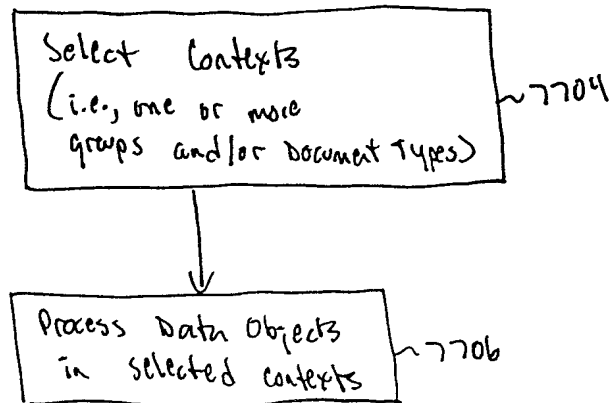
7702
↓

FIG. 77

77/99

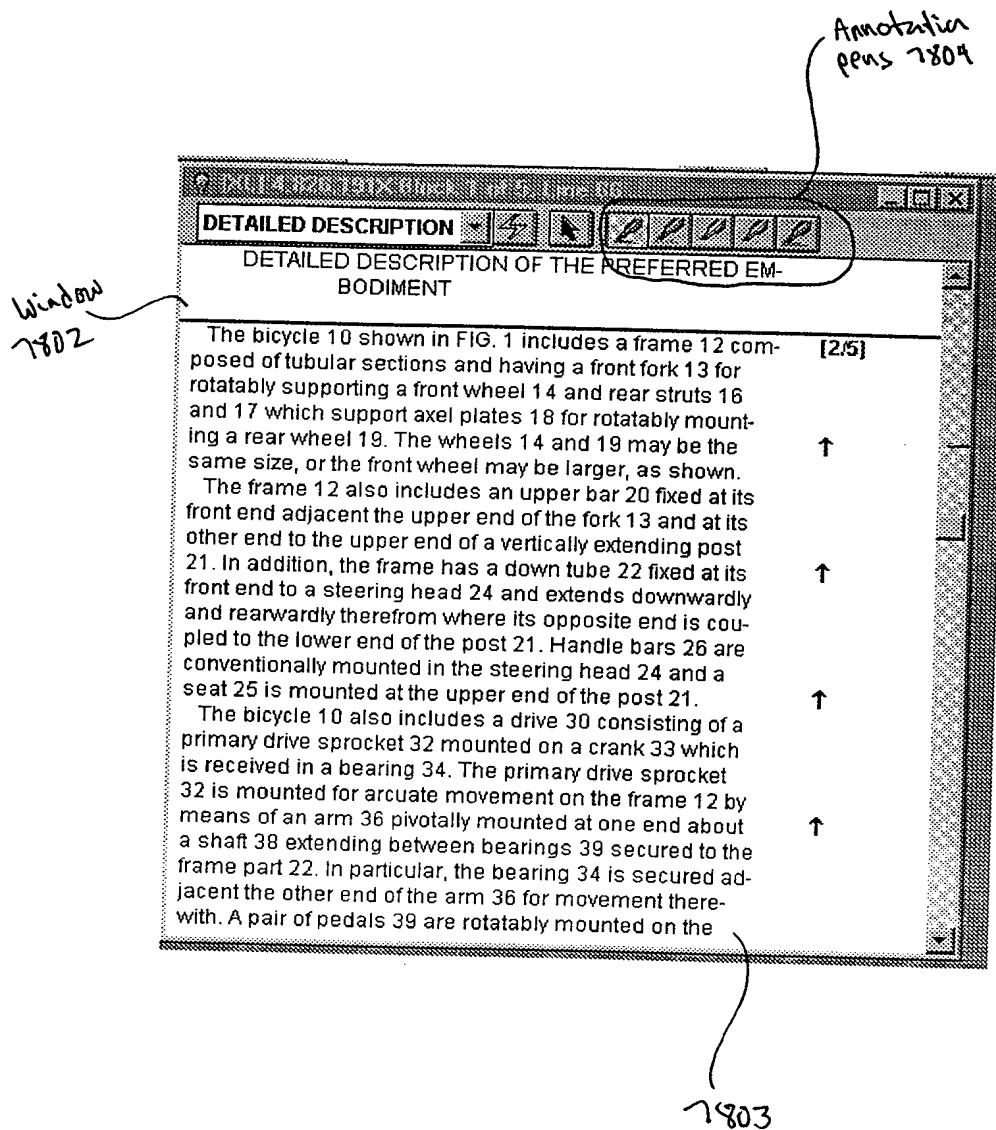


FIG. 78A

78/99

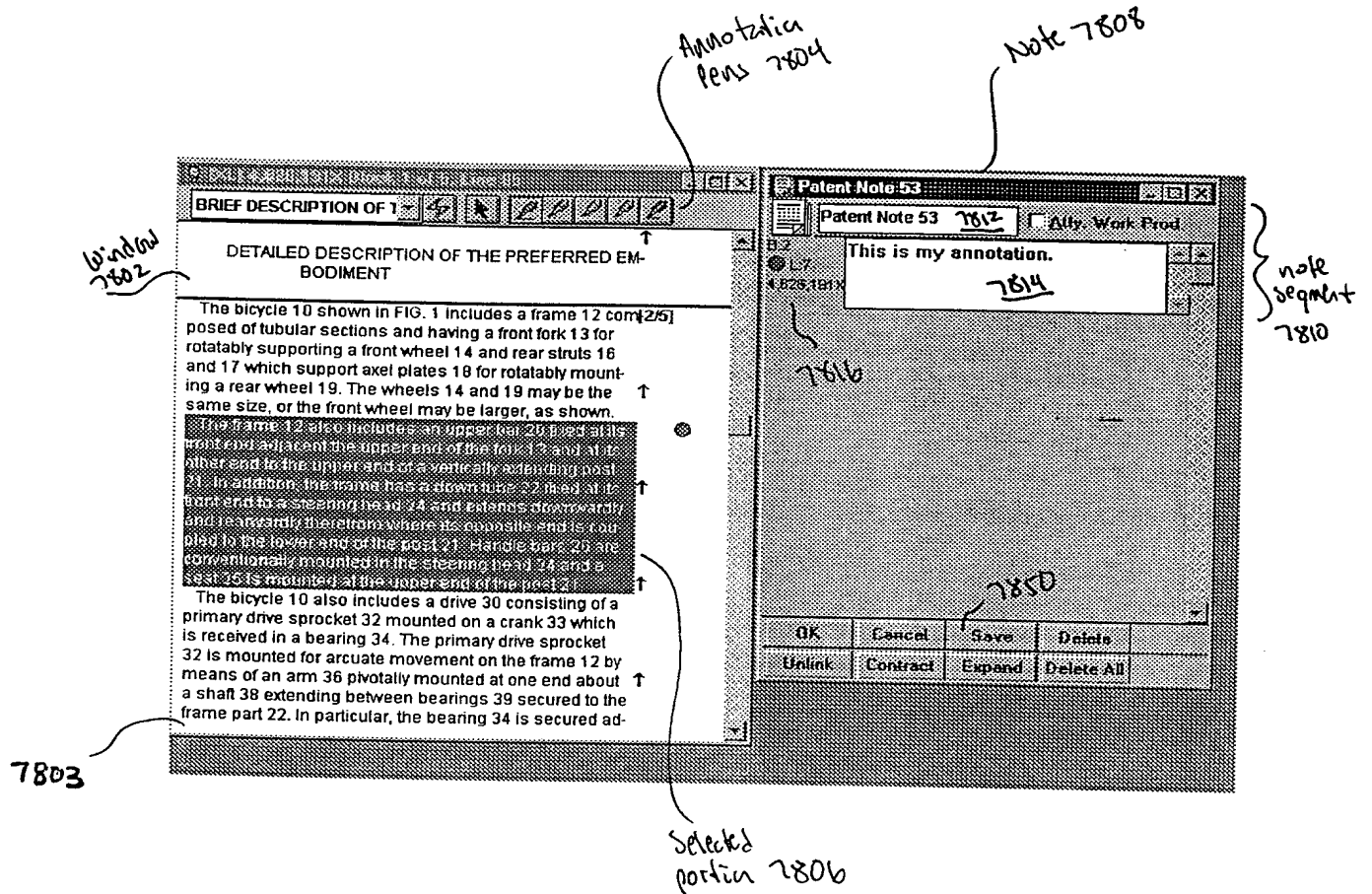


FIG. 78B

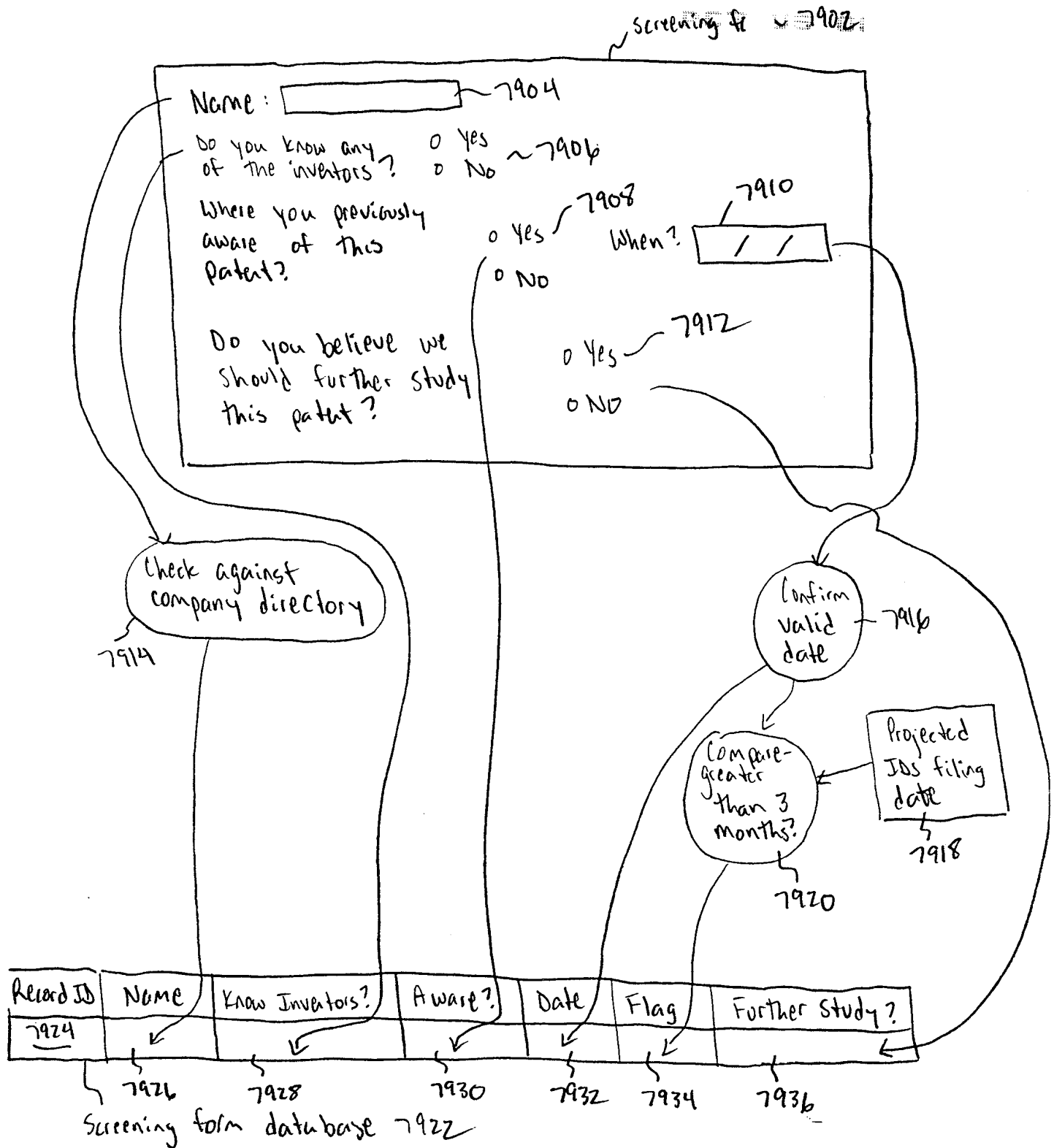
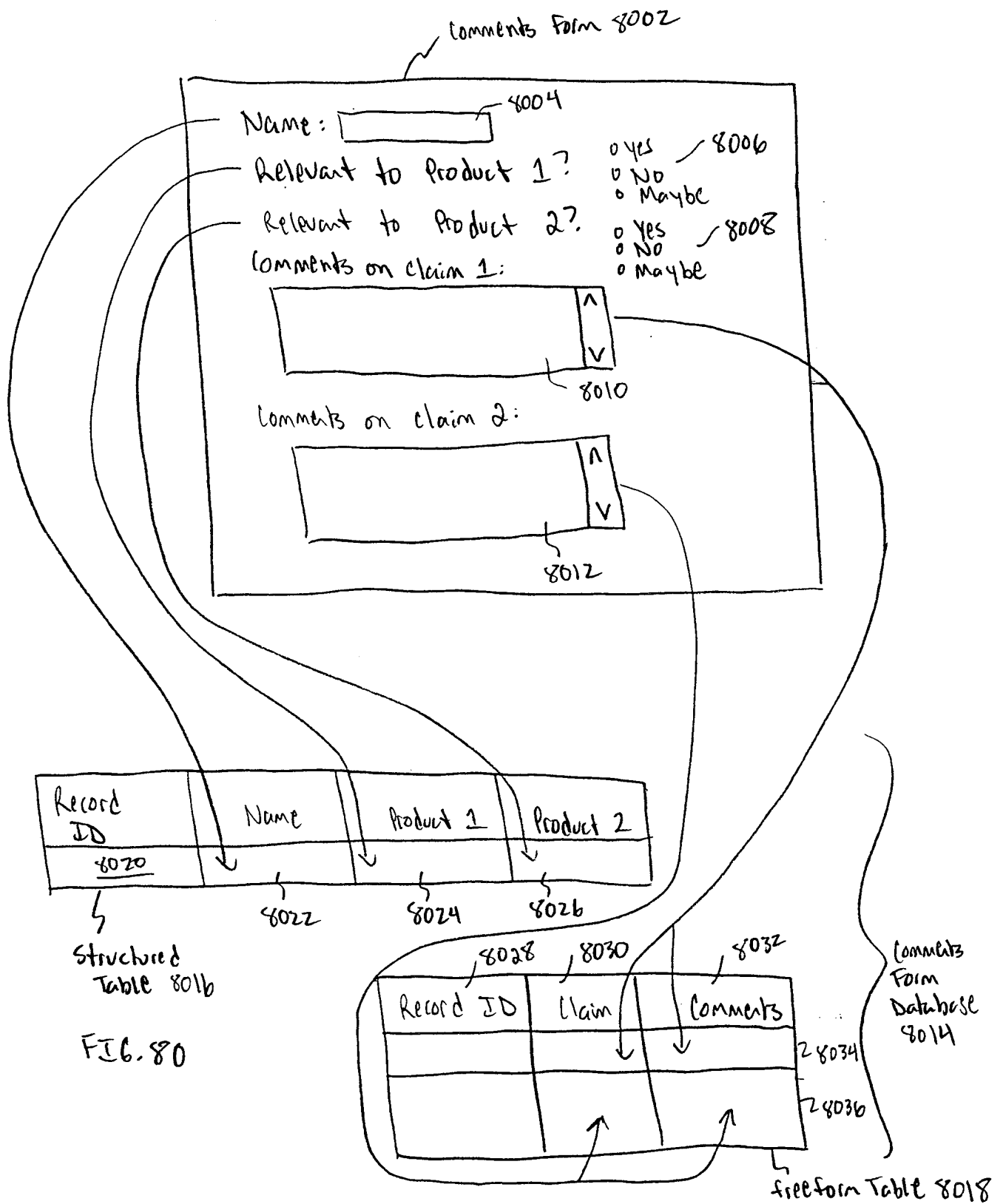


FIG. 79



8/1/99

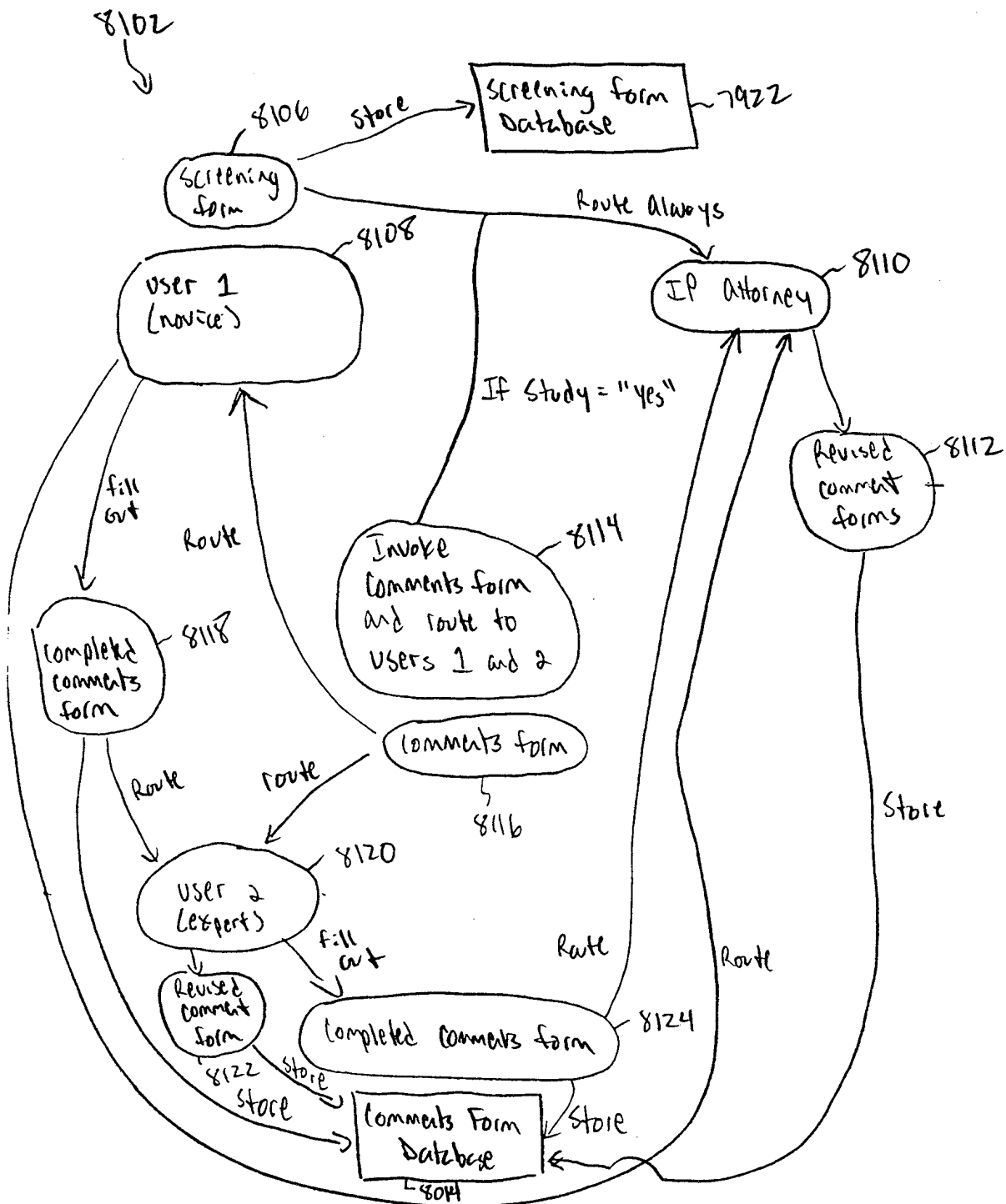


FIG. 81

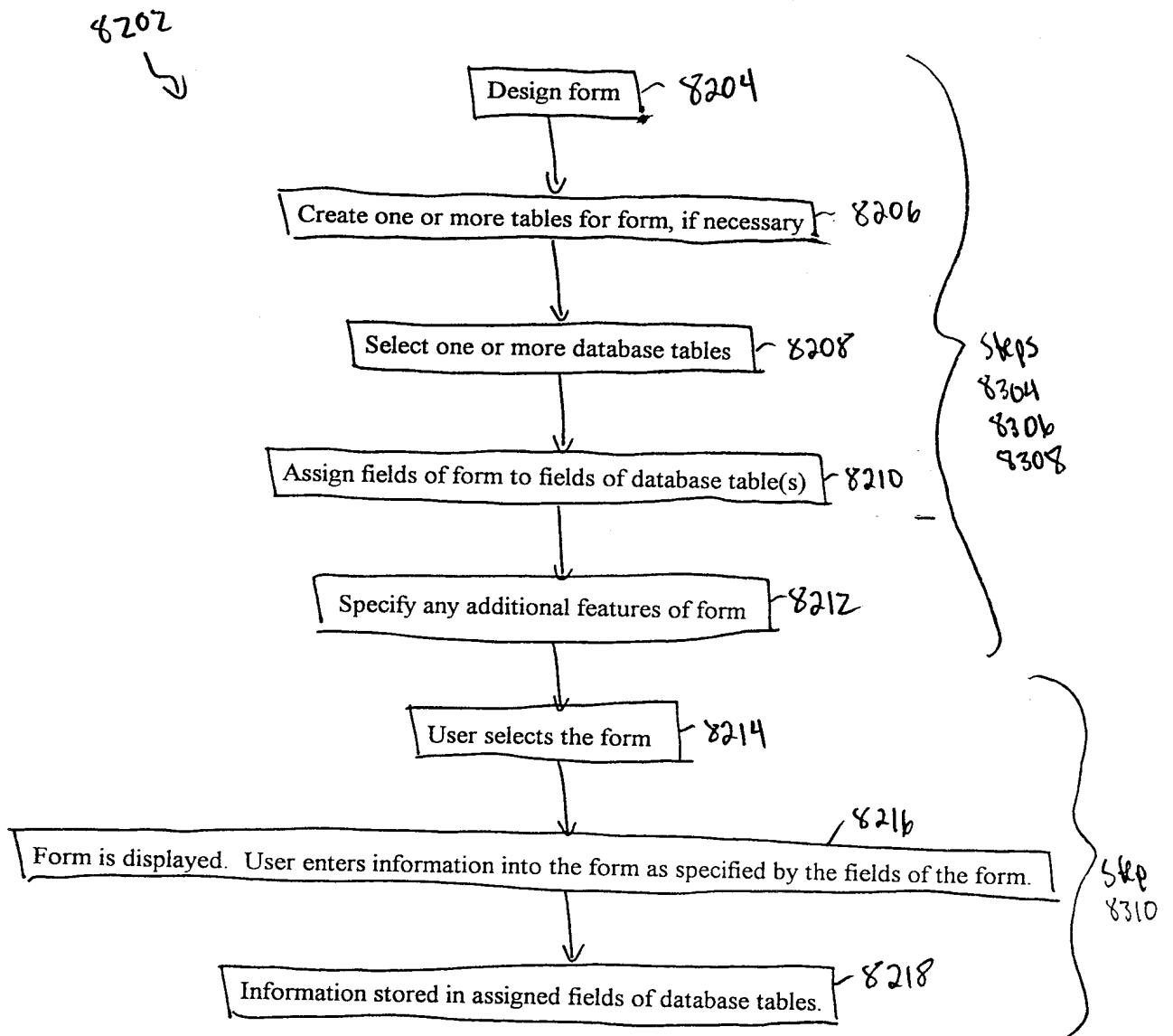


FIG. 82

8302



8304

The system administrator authorizes a given user to create forms. In other words, the ability to create, modify, edit, and otherwise manage forms is preferably a secured operation. This is because the creation of a form will include appropriate provisions for storing the data acquired through the use of a form in a relational database for back-end statistical processing.



8306

The form-creator logs into the system and creates a form to capture the data they are interested in capturing. The invention supports several different types of form input widgets, including radio buttons (choose one from many items), text input fields, check boxes (represents "Yes/No" or "True/False") and so forth. The form-creator explicitly or implicitly associates input widgets with the back-end database tables stored on the server. Even the meta-content of the tables may be stored with the form, allowing for the use of sophisticated data mining applications of the form and the data. It is also possible to associate programmatic operations with the form, thereby allowing the form-creator to control what the annotation-creator sees while the annotation-creator is using the form. Examples of such operations would include "edit checks" on input fields (e.g., the form-creator could specify that an annotation-creator must enter in a number between 0 and 100 when entering data into the form).



FIG. 83A

84/99

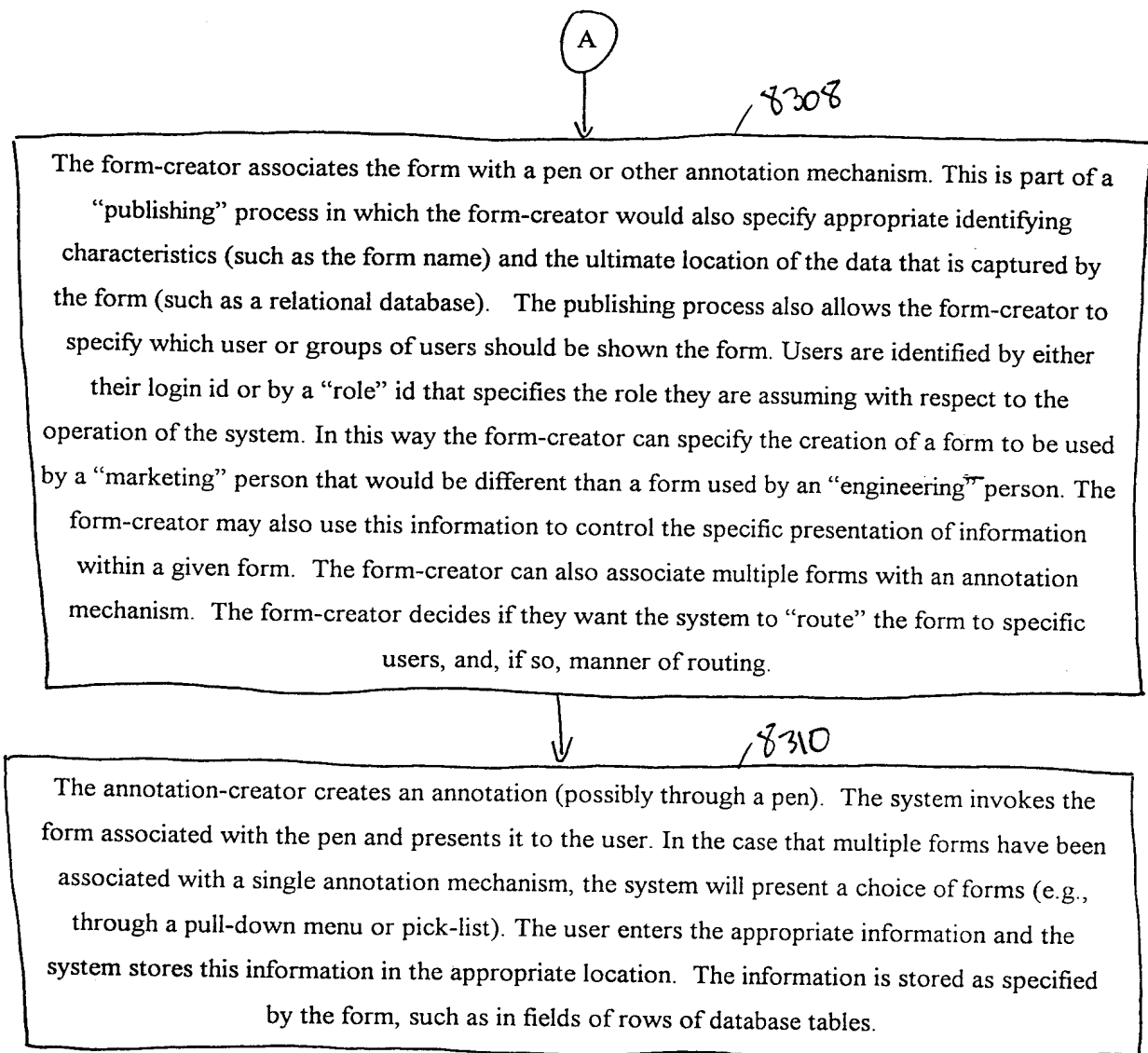


FIG. 83B

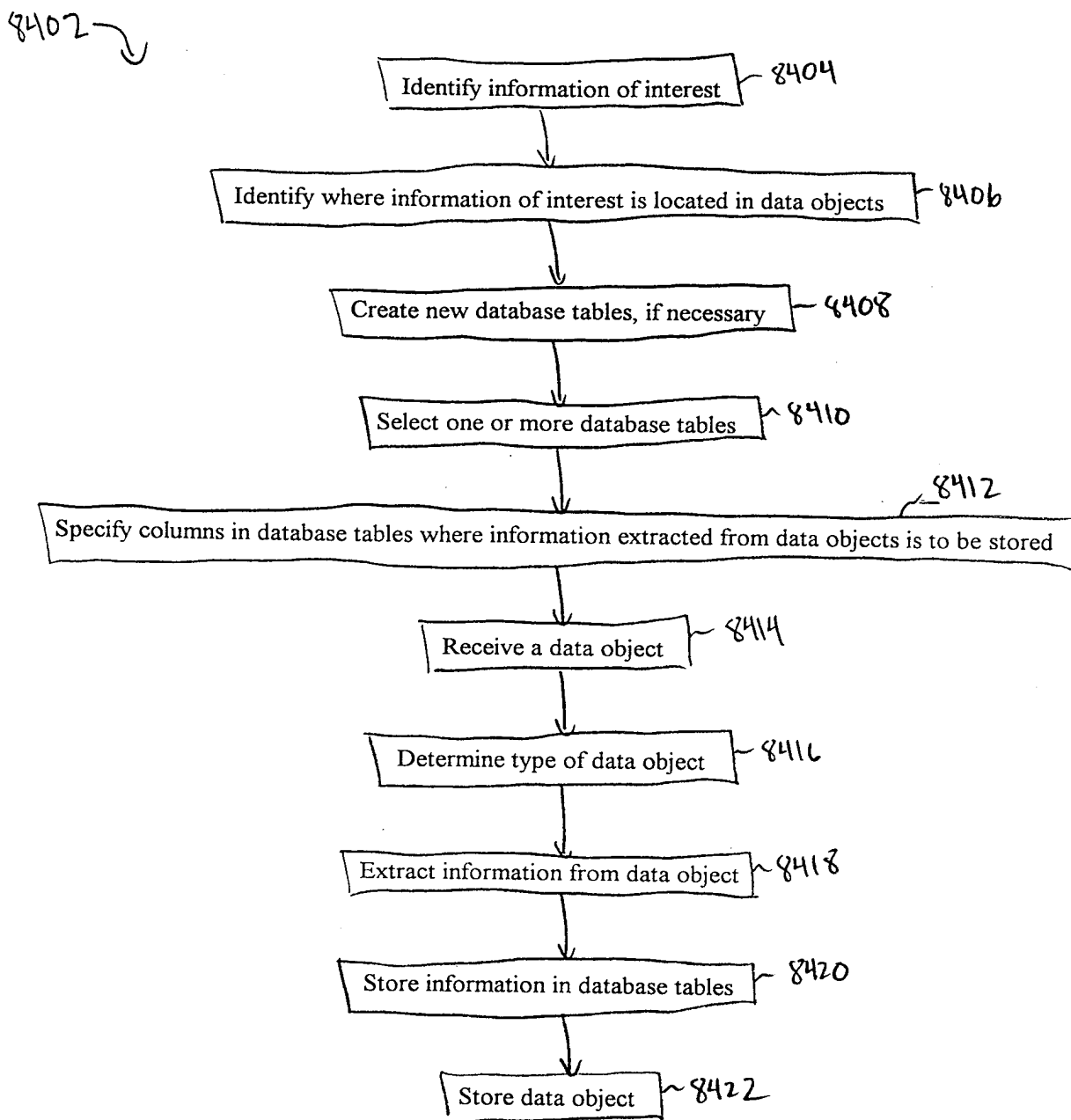


FIG. 84

86/99

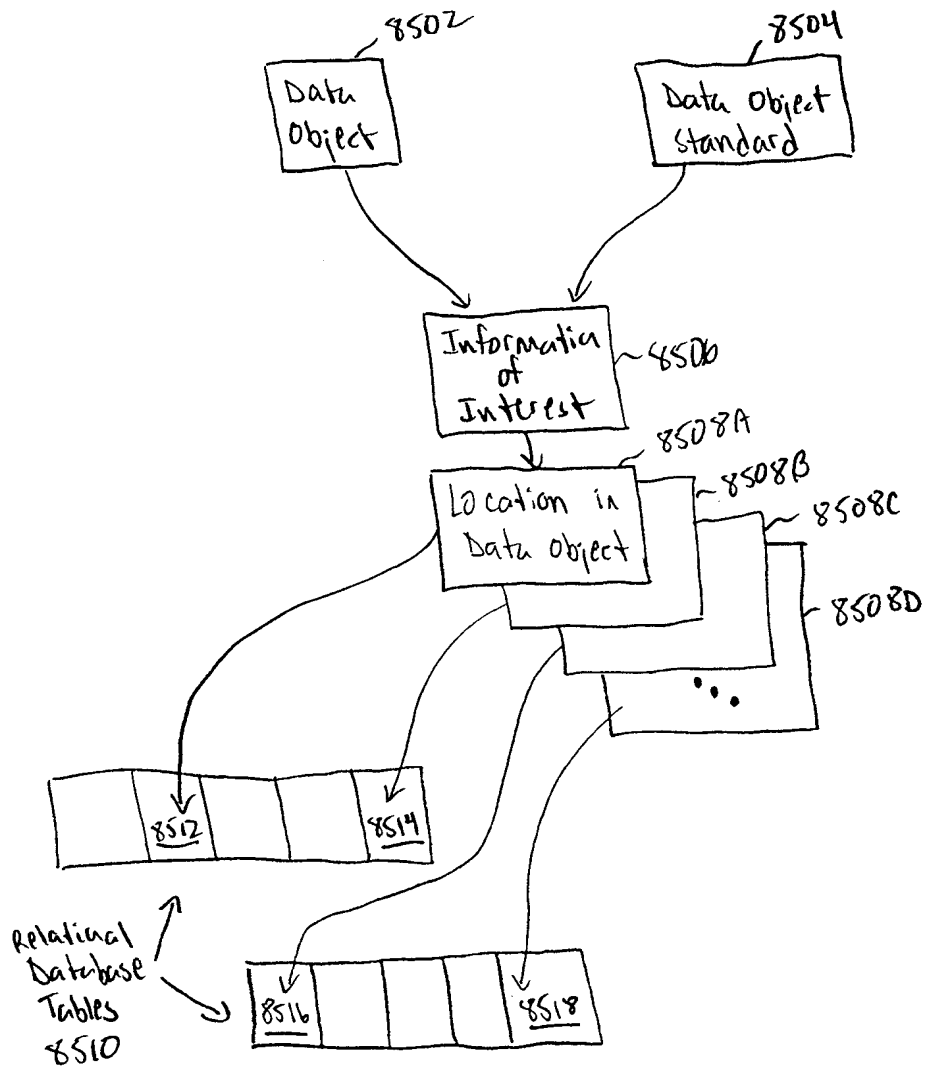


FIG. 85

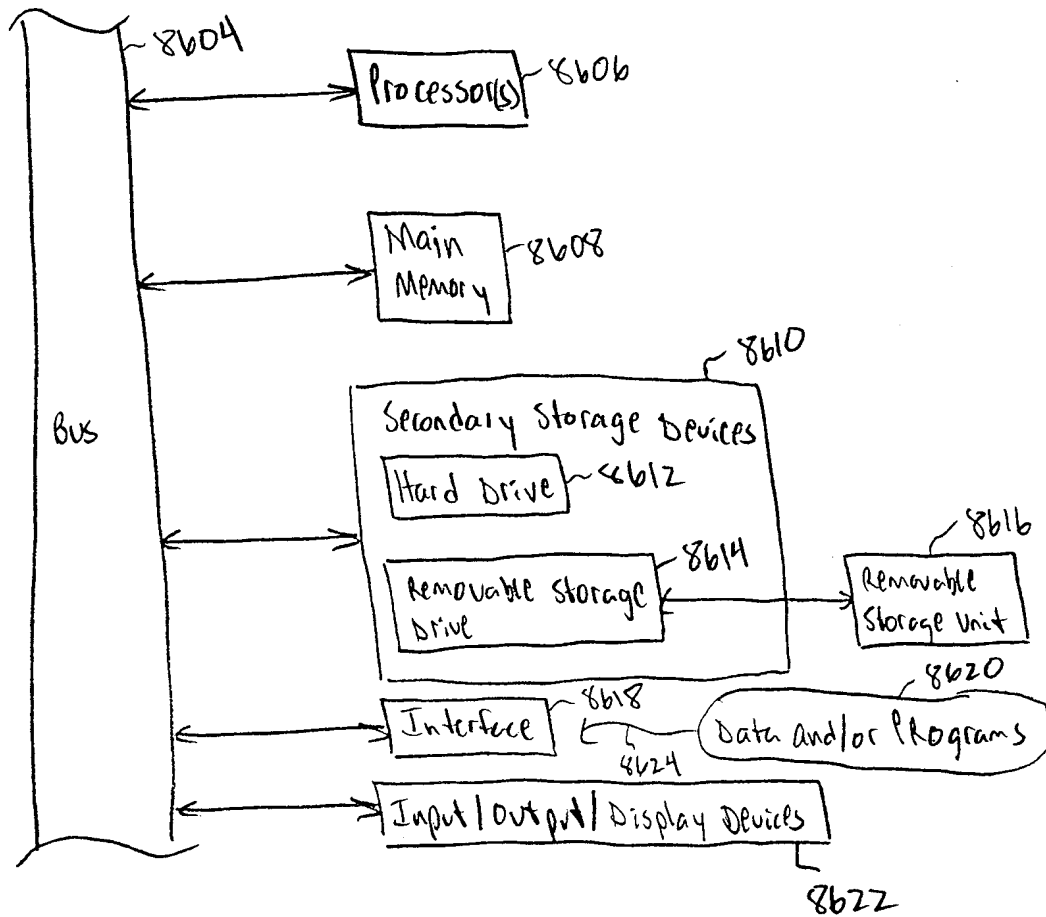
8602
↓

FIG. 8b

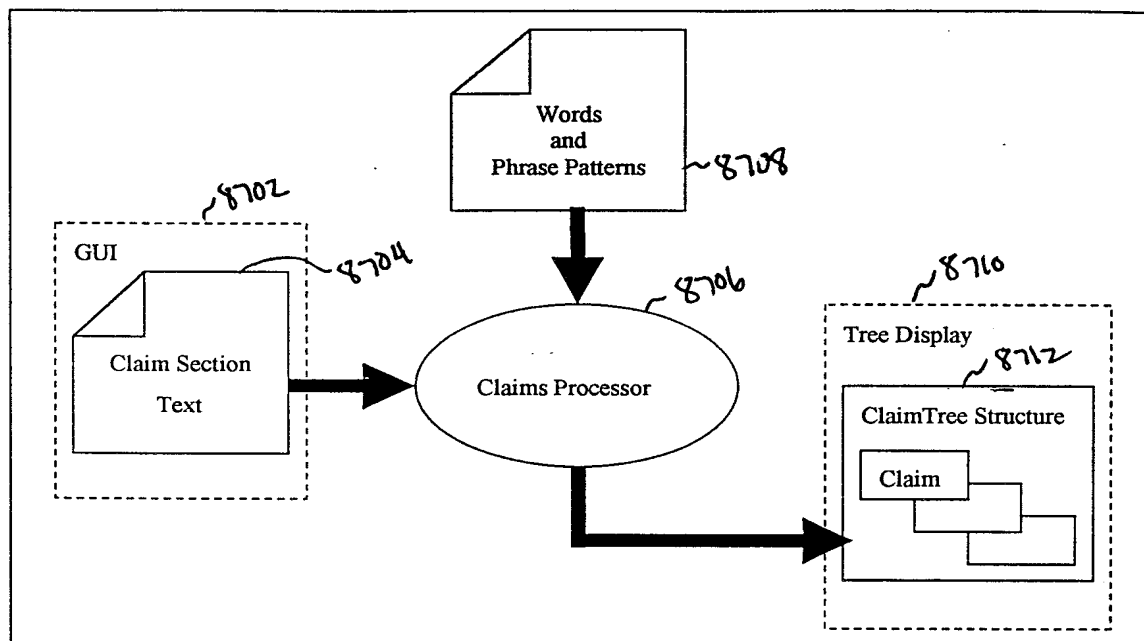
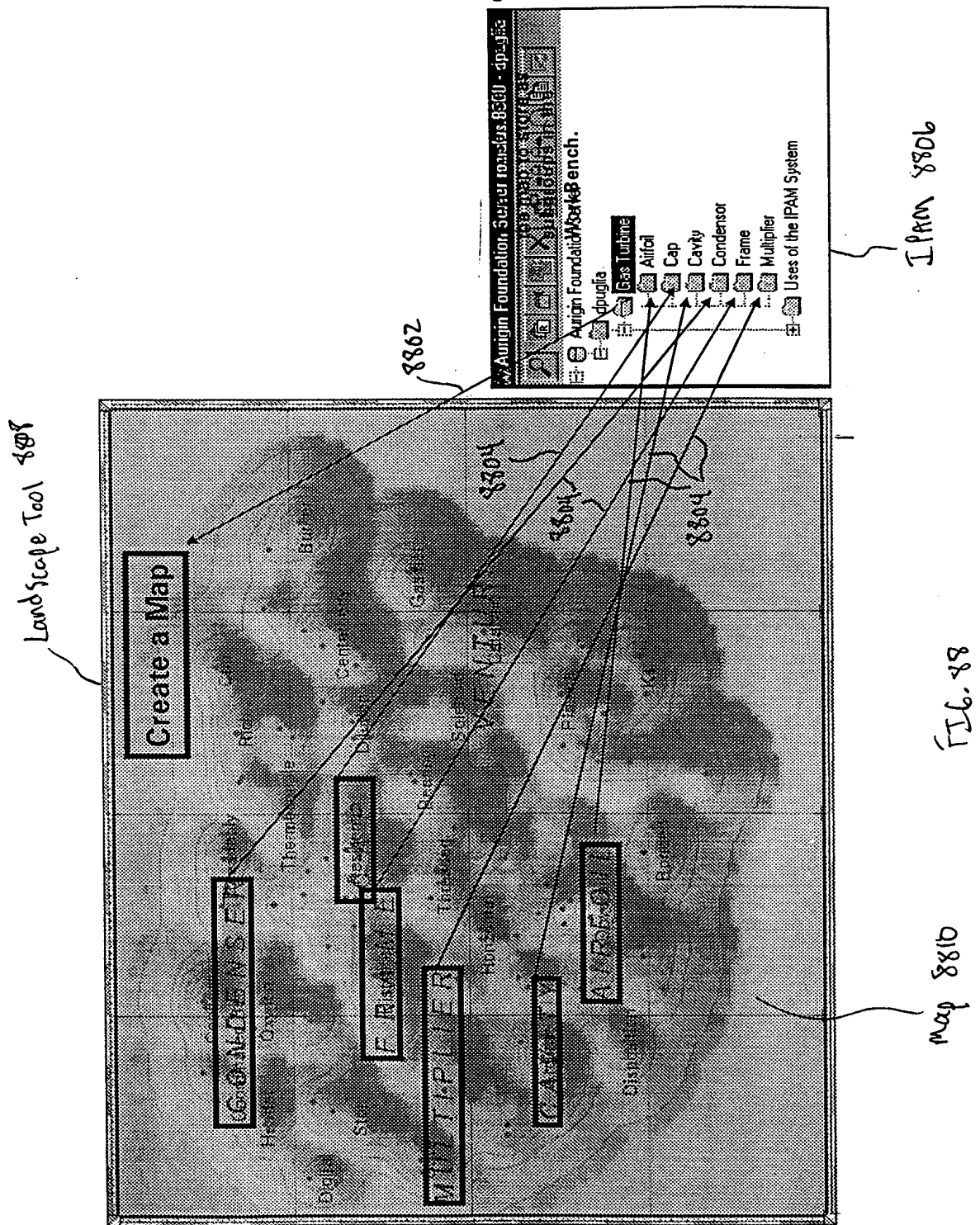
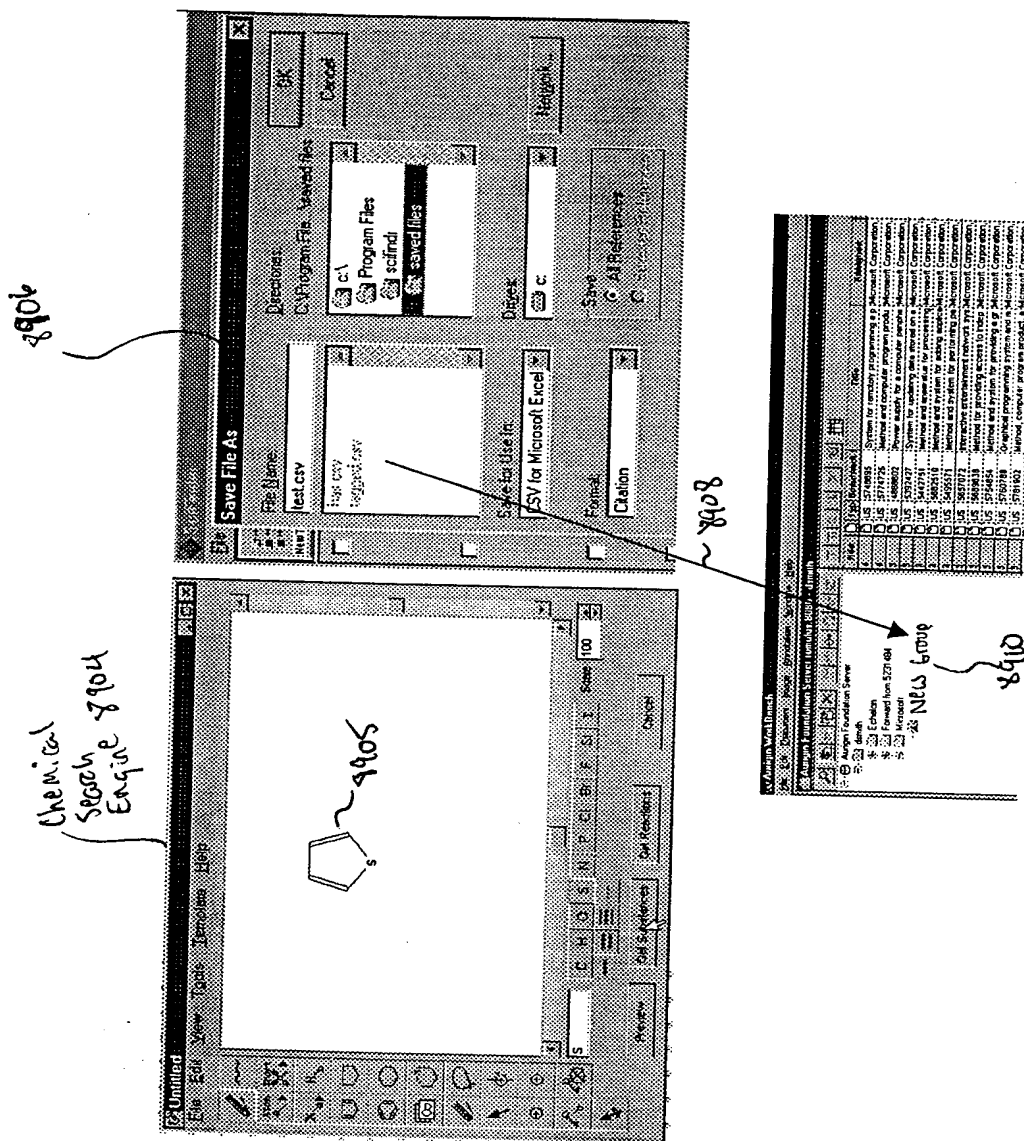


FIG. 87



90/99



ILAM 8907

FIG. 89

91/99

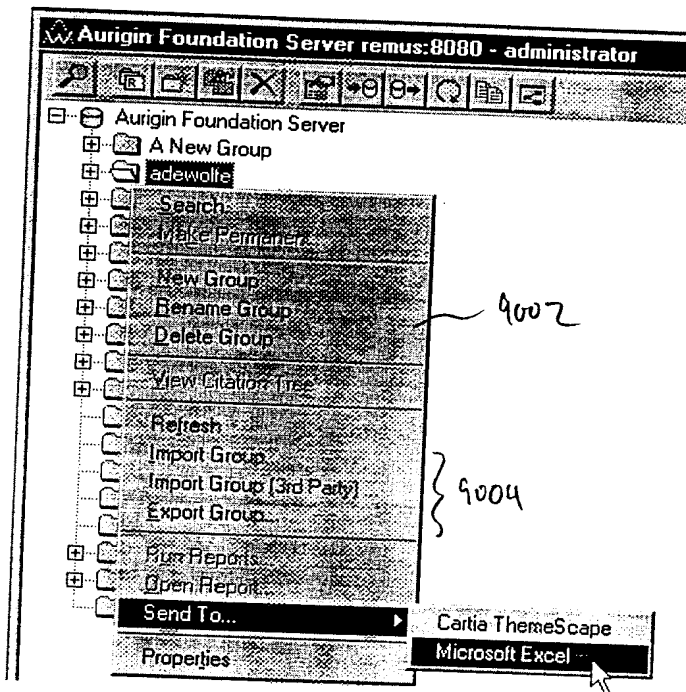


FIG. 90

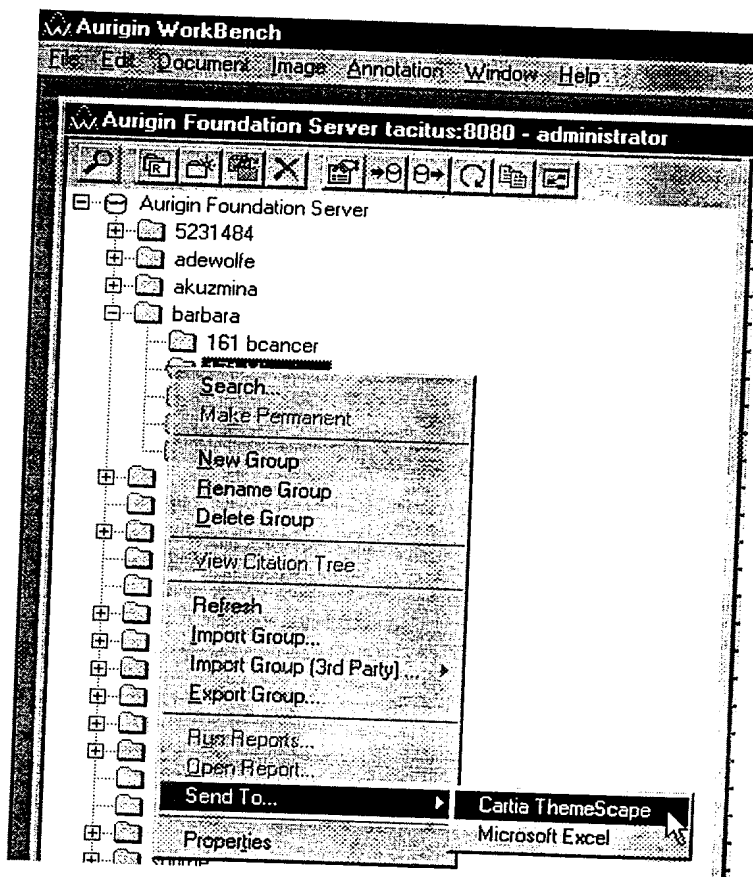


FIG. 91

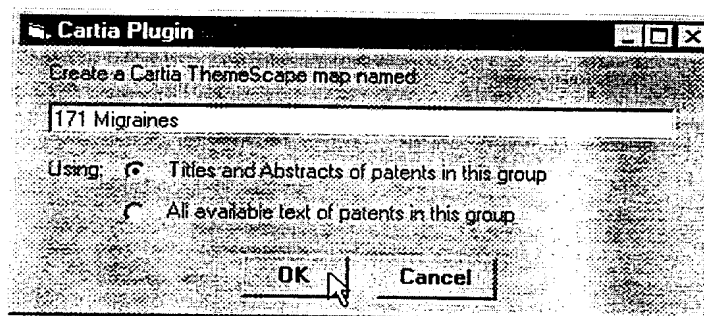


FIG. 92

94/99

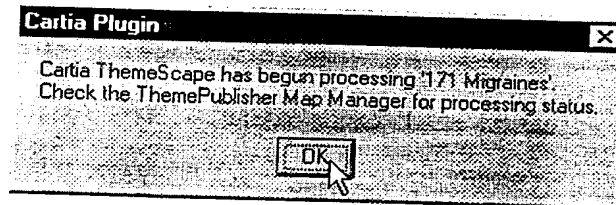


FIG. 93

95/99

Map Manager

Server: pegasus

Email: dsmith@aurigin.com Password: **

Pub	Map Name	# Docs	Size	Status
	171 Migraines	0	600kb	Harvesting item 24: http://taci...
	adewolfe	0	580kb	Ready to Process Too Few ...
	calicheamicin	62	1mb	Processing Complete

Remove Process Public Users Permissions

FI 6. 94

97/99

Document Viewer			
<div>Back Forward Group Circle Contour Clump Select Find Export Print</div>			
Summary		Documents	
Topics	%	Type	Date
Inhibiting	100	EP 0 064 646 A1	4/23/1982
Inhibiting...	100	EP 0 064 646 A1 Title: USE OF NADOLOL FOR TREATING	
A2 Ep	57	• EP 0 331 803 A2	12/2/1988
Nadolol	42	EP 0 331 803 A2 Title: METHOD FOR INHIBITING ONSET OF OR	
Receptor...	28	• EP 0 350 080 A2	4/23/1982
Inhibitor	28	EP 0 350 080 A2 Title: USE OF	
Blocker	28		
A1 Ep	14		
7 of 7 Documents in contour			

FI 6. 96

98/99

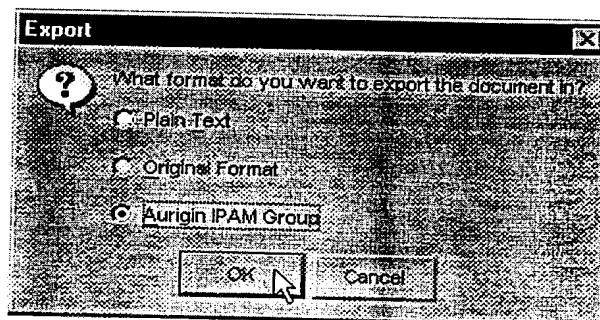


FIG. 97

99/99

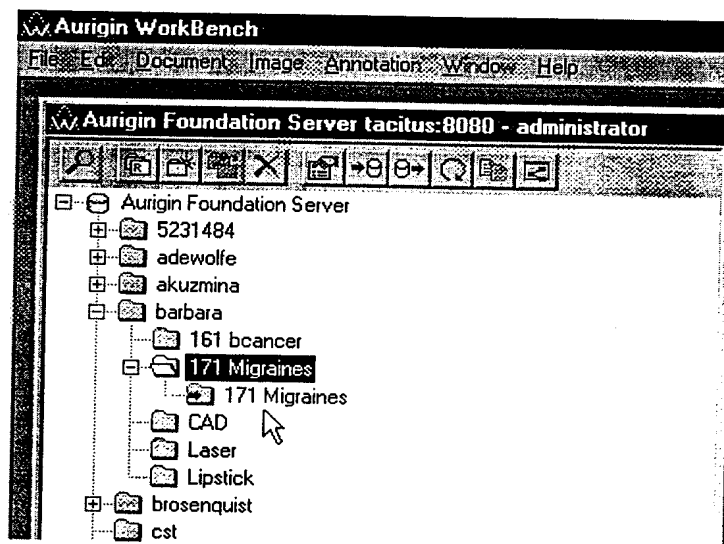


FIG. 98